Journal of Machine Learning Research 17 (2016) 1-37 Submitted 5/14; Revised 11/15; Published 2/16

Learning Taxonomy Adaptation in Large-scale Classification

Rohit Babbar * FIRSTNAME.LASTNAME@TUEBINGEN.MPG.DE
Max-Planck Institute for Intelligent Systems
Tübingen, Germany

Ioannis Partalas * IOANNIS.PARTALAS@VISEO.COM
Viseo Research Center
Grenoble, France

Eric Gaussier
Massih-Reza Amini
Cécile Amblard FIRSTNAME.LASTNAME@IMAG.FR
LIG, Université Grenoble Alpes - CNRS
Grenoble, cedex 9, France, 38041

Editor: Samy Bengio

Abstract

In this paper, we study flat and hierarchical classification strategies in the context of large-scale taxonomies. Addressing the problem from a learning-theoretic point of view, we first propose a multi-class, hierarchical data dependent bound on the generalization error of classifiers deployed in large-scale taxonomies. This bound provides an explanation to several empirical results reported in the literature, related to the performance of flat and hierarchical classifiers. Based on this bound, we also propose a technique for modifying a given taxonomy through pruning, that leads to a lower value of the upper bound as compared to the original taxonomy. We then present another method for hierarchy pruning by studying approximation error of a family of classifiers, and derive from it features used in a meta-classifier to decide which nodes to prune. We finally illustrate the theoretical developments through several experiments conducted on two widely used taxonomies.

Keywords: Large-scale classification, Hierarchical classification, Taxonomy adaptation, Rademacher complexity, Meta-learning

1. Introduction

With the rapid surge of digital data in the form of text and images, the scale of problems being addressed by machine learning practitioners is no longer restricted to the size of training and feature sets, but is also being quantified by the number of target classes. Classification of textual and visual data into a large number of target classes has attained significance particularly in the context of *Big Data*. This is due to the tremendous growth in data from various sources such as social networks, web-directories and digital encyclopedia. Directory Mozilla, DMOZ (`www.dmoz.org`), Wikipedia and Yahoo! Directory (`www.dir.yahoo.com`) are instances of such large scale textual datasets which consist of millions of

∗. Most of this work was done when the authors were affiliated with LIG

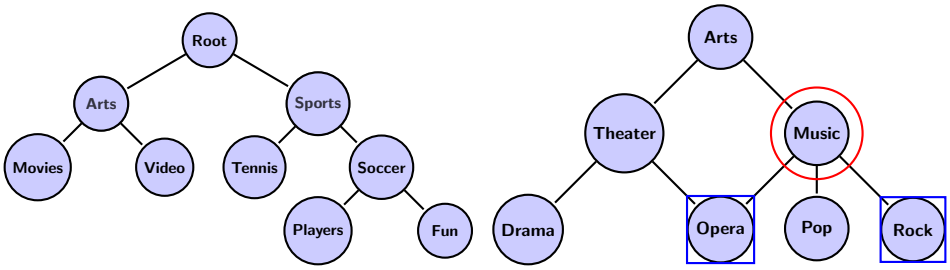

Figure 1: DMOZ and Wikipedia Taxonomies

documents that are distributed among hundreds of thousand target categories. Directory Mozilla, for instance, lists over 5 million websites distributed among close to 1 million categories, and is maintained by close to 100,000 editors. In the more commonly used Wikipedia, which consists of over 30 million pages, documents are typically assigned to multiple categories which are shown at the bottom of each page. The Medical Subject Heading (MESH) [1] hierarchy of the National Library of Medicine is another instance of a large-scale classification system in the domain of life sciences. The target classes in such large-scale scenarios typically have an inherent hierarchical structure among themselves. DMOZ is in the form of a rooted tree where a traversal of path from root-to-leaf depicts transformation of semantics from generalization to specialization. More generally parent-child relationship can exist in the form of directed acyclic graphs, as found in taxonomies such as Wikipedia. The tree and DAG relationship among categories is illustrated for DMOZ and Wikipedia taxonomies in Figure 1.

Due to the sheer scale of the task of classifying data into target categories, there is a definite need to automate the process of classification of websites in DMOZ, encyclopedia pages in Wikipedia and medical abstracts in the MESH hierarchy. However, the scale of the data also poses challenges for the classical techniques that need to be adapted in order to tackle large-scale classification problems. In this context, one can exploit the taxonomy of classes as in the divide-and-conquer paradigm in order to partition the input space. Various classification techniques have been proposed for deploying classifiers in such large-scale scenarios, which differ in the way they exploit the given taxonomy. These can be broadly divided into four main categories :

- *Hierarchical top-down strategy* with independent classification problems at each node,
- Hierarchical top-down strategy on a *simplified hierarchy*, such as by partially flattening the hierarchy,
- Ignoring the hierarchy information altogether and using *flat classification* that is, training one classifier for each target class, and
- Using the taxonomy information for an *appropriate loss-function design* such as by considering the distance between the true and predicted target class label.

In large-scale classification involving tens of thousand of target categories, the goal of a machine learning practitioner is to achieve the best trade-off among the various metrics

1. `https://www.nlm.nih.gov/mesh/`

of interest. Flat and top-down hierarchical classification perform significantly differently on these metrics of interest which primarily include prediction accuracy, computational complexity of training, space complexity of the trained model and complexity of prediction.

1.1 Prediction Accuracy

Our focus in this work is primarily on the prediction accuracy when dealing with large-scale category systems. Hierarchical models for large scale classification, such as top-down Pachinko-machine based methods, suffer from the drawback that they have to make many decisions prior to reaching a final category, which leads to the error propagation phenomenon causing a decrease in accuracy. This is mainly due to the fact that the top level classes in large scale taxonomies are quite general. For example, *Business* and *Shopping* categories in DMOZ (not shown in Figure 1 above) are likely to be confused while classifying a new document. Moreover, since the classification is not recoverable, it leads to the phenomenon of error propagation and hence degrades accuracy at the leaf level. On the other hand, flat classifiers rely on a single decision including all the final categories, a single decision that is however difficult to make as it involves many categories, which are potentially unbalanced. It is thus very difficult to assess which strategy is better and there is no consensus, at the time being, on to which approach, flat or hierarchical, should be preferred on a particular category system. Furthermore, we explore the methods which learn to adapt the given hierarchy of categories such that the resulting hierarchy leads to better classification in the top-down Pachinko-machine based method. We also show that when dealing with large number of power-law distributed categories, taxonomy adaptation by pruning some nodes leads to better classification than building taxonomies from scratch.

1.2 Computational Complexity of Training and Prediction

When dealing with large-scale category systems, flat and hierarchical classification techniques exhibit significant difference in their performance when compared on the basis of computational complexity of training and prediction. In the pioneering work of Liu et al. (2005), an extensive comparison of training time complexity for flat and hierarchical classification has been studied. Using the power-law distribution of documents among categories, the authors analytically and empirically demonstrate that the training time of top-down Pachinko machine classification strategy is orders of magnitude better than that for flat classification.

In terms of complexity of prediction for flat classification when dealing with K target categories, for every test instance one needs to evaluate the inner-product with $O(K)$ weight vectors in . This is much higher than the logarithmic computational complexity of prediction in a top-down Pachinko-machine where only $O(\log(K))$ weight-vectors need to be evaluated. In view of this advantage for tree-based classifiers, there has been a surge in research works on the techniques for automatically building taxonomy of target categories (Bengio et al., 2010; Gao and Koller, 2011; Deng et al., 2011; Agrawal et al., 2013). The focus in these works is to show that by building such tree-based taxonomy of categories, one can reduce the complexity of prediction, while still maintaining good rates for accuracy of prediction.

Since the computational complexity of training and prediction has been studied in the above works, and it has been already shown that top-down methods are more favorable

as compared to flat method, we do not focus on these aspects explicitly in this paper. Furthermore, in our recent work Babbar et al. (2014a), we present a quantitative analysis of the fit to power-law distribution of documents among categories in large-scale category systems, and show that space complexity of top-down classification methods is lower than that of flat methods under conditions that typically hold in practice.

2. Contributions and Related Work

2.1 Contributions

One of the research challenges we address in this work is the study of flat versus hierarchical classification in large-scale taxonomies from a learning-theoretic point of view, and the consistency of the empirical risk minimization principle for the Pachinko-machine based methods. This theoretical analysis naturally leads to a model selection problem. We extend and elaborate further on our recent work Babbar et al. (2013a) to address the problem of choosing between the two strategies. We introduce bounds based on Rademacher complexity for the generalization errors of classifiers deployed in large-scale taxonomies. These bounds explicitly demonstrate the trade-off that both flat and hierarchical classifiers face in large-scale taxonomies.

Even though the given human-built taxonomies such as DMOZ and Yahoo! Directory provide a good starting point to capture the underlying semantics of the target categories, these may not be optimal, especially in the presence of large number of power-law distributed categories. In the second part of our contributions, we then propose a strategy for taxonomy adaptation which modifies the given taxonomy by pruning nodes in the tree to output a new taxonomy which is better suited for the classification problem. In this part, we exploit the generalization error bound developed earlier to build a criterion for Support Vector Machine classifier, so as to choose the nodes to prune in a computationally efficient manner. We also empirically show that adapting a given taxonomy leads to better generalization performance as compared to the original taxonomy and the ones obtained by taxonomy construction methods (Beygelzimer et al., 2009b; Choromanska and Langford, 2014). This difference is particularly magnified in category systems in which categories are power-law distributed. As discussed extensively in the work of Liu et al. (2005), power-law distribution of documents among categories is a common phenomenon in most naturally occurring category systems such as Yahoo! directory and Directory Mozilla.

In the third part, we present a more comprehensive approach for pruning the given hierarchy that is applicable to both discriminative and generative classifiers. Towards this end, we cover the Logistic Regression and Naive Bayes classifiers and present approximation error based bounds for their multi-class versions. Based on these bounds, we then propose a meta-learning strategy for hierarchy pruning applicable for both discriminative and generative classifiers. We perform a three-step procedure towards achieving hierarchy pruning, (i) we make classifier specific theoretical analysis to identify the key features which determine the variation in classification accuracy upon flattening, (ii) based on the features obtained in the step above, we train a meta-classifier on a validation set, and finally, (iii) the meta-classifier when presented with an unseen hierarchy and the corresponding training data modifies it to output the desired hierarchy which leads to better classification accuracy. Contrary to Dekel (2009) that reweighs the edges in a taxonomy through a cost sensitive

loss function to achieve this goal, our simple pruning strategy modifies the taxonomy in an explicit way.

Lastly, we empirically demonstrate and verify the theoretical findings for flat and hierarchical classification on several large-scale taxonomies extracted from DMOZ and the International Patent Classification (IPC) collections. The experimental results are in line with results reported in previous studies, as well as with our theoretical developments. Secondly, we also study the impact of the two methods proposed for taxonomy adaptation by pruning the hierarchy. Lastly, these strategies are also empirically compared against those that build taxonomies from scratch such as LOMTree (Choromanska and Langford, 2014) and FilterTree (Beygelzimer et al., 2009b).

2.2 Related Work

Large-scale classification, involving tens of thousand target categories, has assumed significance in the era of Big data. Many approaches for classification of data in large number of target categories have been proposed in the context of text and image classification. These approaches differ in the manner in which they exploit the semantic relationship among categories. In similar vein, open challenges such as Large-scale Hierarchical Text Classification (LSHTC) (Partalas et al., 2015) and Large Scale Visual Recognition Challenge (ILSVRC)[2] (Russakovsky et al., 2014) have been organized in recent years.

Some of the earlier works on exploiting hierarchy among target classes for the purpose of text classification has been studied by Koller and Sahami (1997) and Dumais and Chen (2000). These techniques use the taxonomy to train independent classifiers at each node in the top-down Pachinko Machine manner. Parameter smoothing for Naive Bayes classifier along the root to leaf path was explored by McCallum et al. (1998). The work by Liu et al. (2005) is one of first studies to apply hierarchical SVM to the scale with over 100,000 categories in Yahoo! directory. More recently, other techniques for large scale hierarchical text classification have been proposed. Prevention of error propagation by applying *Refined Experts* trained on a validation was proposed by Bennett and Nguyen (2009). In this approach, bottom-up information propagation is performed by utilizing the output of the lower level classifiers in order to improve the classification of top-level classifiers. Another approach to control the propagation of error in tree-based classifiers is to explore multiple root-to-leaf paths as in beam-search (Norvig, 1992). In this respect, Fleuret and Geman (2001); Sun et al. (2013) proposed such approaches. However, this increases the computationally complexity of prediction especially in the presence of large-number of target categories and hence these methods may not scale well for tens of thousand target categories. Deep Classification by Xue et al. (2008) proposes hierarchy pruning to first identify a much smaller subset of target classes. Prediction of a test instance is then performed by re-training a Naive Bayes classifier on the subset of target classes identified from the first step.

Using the taxonomy in the design of loss function for maximum-margin based approaches have been proposed by Cai and Hofmann (2004); Dekel et al. (2004), where the degree of penalization in mis-classification depends on the distance between the true and predicted class in the hierarchy tree. Another recent approach by Dekel (2009) which proposes to make the loss function design robust to class-imbalance and arbitrariness problems in taxonomy

2. http://www.image-net.org/challenges/LSVRC/

structure. However, these approaches were applied to the datasets in which the number of categories were limited to a few hundreds. Bayesian modeling of large scale hierarchical classification has been proposed by Gopal et al. (2012) in which hierarchical dependencies between the parent-child nodes are modeled by centering the prior of the child node at the parameter values of its parent. Recursive-regularization based strategy for large-scale classification has been proposed by Gopal and Yang (2013). The approaches presented in the two studies above attempt to solve the problem wherein the number of categories are in the range of tens of thousands. In both these works, the authors employ the intuition that the weight vectors of a parent-child pair of nodes should be close to each other. This can be enforced in the form of a prior in the Bayesian approach (Gopal et al., 2012) and as a regularizer in the recursive regularization approach (Gopal and Yang, 2013). However, on most of the large-scale datasets used in these papers, the accuracy performance (Micro-F1) of the proposed approaches is close to the flat classification scheme for which ready to use packages such as Liblinear are available. Another study related to our work is the one of Narasimhan et al. (2015) that studies the consistency of hierarchical classification algorithms with respect to the tree distance metric on the hierarchy tree of class labels.

Hierarchy simplification by flattening entire layer in the hierarchy has been studied from an empirical view-point by Wang and Lu (2010); Malik (2009). These strategies do not provide any theoretical justification for applying this procedure. Moreover, they offer no clear guidelines regarding which layer in the hierarchy one should flatten. In contrast, our strategy presented in this paper for taxonomy adaptation has the advantage that, (i) it is based on a well-founded theoretical criteria, and (ii) it is applied in a node-specific sense rather than to an entire layer. This strategy is also similar in spirit to the approach presented in our another recent study Babbar et al. (2013b) which is motivated from Perceptron Decision Trees (Bennett et al., 2000). The study by Weinberger and Chapelle (2008) introduces a slightly different simplification of the hierarchy of classes, and it achieves this by an embedding the classes and documents into a common space. Our recent work Babbar et al. (2013b) for hierarchy simplification by pruning nodes in a large-scale taxonomy is similar in spirit to the approach presented in this paper. Semi-supervised approach for hierarchical classification in incomplete hierarchies has been presented in the recent work of Dalvi and Cohen (2014). A post-processing approach for improving rare categories detection in large-scale power-law distributed category systems is discussed in the work by Babbar et al. (2014b).

Apart from accuracy, other important factors while evaluating the classification strategies for large scale classification are training and prediction speed. Learning the hierarchy tree from large number of classes in order to make faster prediction has also attained significance as explored in the recent works by Bengio et al. (2010); Beygelzimer et al. (2009a); Gao and Koller (2011); Choromanska and Langford (2014). The aim in these approaches is to achieve better prediction speed while maintaining the same classification accuracy as flat classification. On the other end of the spectrum are flat classification techniques such as employed by Perronnin et al. (2012) which ignore the hierarchy structure altogether. These strategies are likely to perform well for balanced hierarchies with sufficient training instances per target class and not so well in *truly* large-scale taxonomies which suffer from the problem of rare categories. In this respect, our work is unique in the sense that by performing selective hierarchy pruning we improve accuracy over the fully hierarchical

strategy. Furthermore, since the proposed pruning method maintains the overall hierarchical structure, it enjoys the computational advantages of better training and prediction speed.

The remainder of the paper is organized as follows: In Section 2.2 we review the recently proposed approaches in the context of large-scale hierarchical text classification. We introduce the notations used in Section 3 and then study flat versus hierarchical strategies by studying the generalization error bounds for classification in large-scale taxonomies. Taxonomy adaptation by pruning the hierarchy using the developed generalization error analyses is given in Section 4. Approximation error for multi-class versions of Naive Bayes and Logistic Regression classifiers are presented in Section 5.1 and Section 5.2 respectively, and the meta-learning based hierarchy pruning method is presented in Section 5.3. Section 6 illustrates these developments via experiments and finally, Section 7 concludes this study.

3. Flat versus Hierarchical Classification

In this section, we present the generalization error analysis for top-down hierarchical classification using the notion of Rademacher complexity for measuring the complexity of a function class. This will motivate a criterion for choosing between flat and hierarchical classification for a given category system. More importantly, the criterion will be based on quantities that can computed from the training data.

3.1 A hierarchical Rademacher data-dependent bound

Let $\mathcal{X} \subseteq \mathbb{R}^d$ be the input space and let V be a finite set of class labels. We further assume that examples are pairs $(\mathbf{x}, v)$ drawn according to a fixed but unknown distribution $\mathcal{D}$ over $\mathcal{X} \times V$. In the case of hierarchical classification, the hierarchy of classes $\mathcal{H} = (V, E)$ is defined in the form of a rooted tree, with a root $\perp$ and a parent relationship $\pi : V \setminus \{\perp\} \to V$ where $\pi(v)$ is the parent of node $v \in V \setminus \{\perp\}$, and E denotes the set of edges with parent to child orientation. For each node $v \in V \setminus \{\perp\}$, we further define the set of its siblings $\mathfrak{S}(v) = \{v' \in V \setminus \{\perp\}; v \neq v' \wedge \pi(v) = \pi(v')\}$ and its children $\mathfrak{D}(v) = \{v' \in V \setminus \{\perp\}; \pi(v') = v\}$. The nodes at the intermediary levels of the hierarchy define general class labels while the specialized nodes at the leaf level, denoted by $\mathcal{Y} = \{y \in V : \nexists v \in V, (y, v) \in E\} \subset V$, constitute the set of target classes. Finally for each class y in $\mathcal{Y}$ we define the set of its ancestors $\mathfrak{P}(y)$ defined as

$$\mathfrak{P}(y) = \{v_1^y, \ldots, v_{k_y}^y; v_1^y = \pi(y) \wedge \forall l \in \{1, \ldots, k_y - 1\}, v_{l+1}^y = \pi(v_l^y) \wedge \pi(v_{k_y}^y) = \perp\}$$

For classifying an example $\mathbf{x}$, we consider a top-down classifier making decisions at each level of the hierarchy, this process sometimes referred to as the *Pachinko* machine selects the best class at each level of the hierarchy and iteratively proceeds down the hierarchy. In the case of flat classification, the hierarchy $\mathcal{H}$ is ignored, $\mathcal{Y} = V$, and the problem reduces to the classical supervised multi-class classification problem.

Our main result is the following theorem which provides a data-dependent bound on the generalization error of a top-down multi-class hierarchical classifier. We consider here kernel-based hypotheses, with $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ a PDS (positive definite symmetric) kernel

and $\Phi : \mathcal{X} \rightarrow \mathbb{H}$ its associated feature mapping function, defined as :

$$\mathcal{F}_B = \{f : (\mathbf{x}, v) \in \mathcal{X} \times V \mapsto \langle \Phi(\mathbf{x}), \mathbf{w}_v \rangle \mid \mathbf{W} = (w_1 \ldots, w_{|V|}), ||\mathbf{W}||_{\mathbb{H}} \leq B\}$$

where $\mathbf{W} = (w_1 \ldots, w_{|V|})$ is the matrix formed by the $|V|$ weight vectors defining the kernel-based hypotheses, $\langle ., . \rangle$ denotes the dot product, and $||\mathbf{W}||_{\mathbb{H}} = \left(\sum_{v \in V} ||\mathbf{w}_v||^2\right)^{1/2}$ is the $L^2_{\mathbb{H}}$ group norm of $\mathbf{W}$. We further define the following associated function class:

$$\mathcal{G}_{\mathcal{F}_B} = \{g_f : (\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y} \mapsto \min_{v \in \mathfrak{P}(y)} (f(\mathbf{x}, v) - \max_{v' \in \mathfrak{S}(v)} f(\mathbf{x}, v')) \mid f \in \mathcal{F}_B\}$$

For a given hypothesis $f \in \mathcal{F}_B$, the sign of its associated function $g_f \in \mathcal{G}_{\mathcal{F}_B}$ directly defines a hierarchical classification rule for f as the top-down classification scheme outlined before simply amounts to: *assign* $\mathbf{x}$ *to* y *iff* $g_f(\mathbf{x}, y) > 0$. The learning problem we address is then to find a hypothesis f from $\mathcal{F}_B$ such that the generalization error of $g_f \in \mathcal{G}_{\mathcal{F}_B}$, $\mathcal{E}(g_f) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left[\mathbb{1}_{g_f(\mathbf{x},y)\leq 0}\right]$, is minimal ($\mathbb{1}_{g_f(\mathbf{x},y)\leq 0}$ is the 0/1 loss, equal to 1 if $g_f(\mathbf{x}, y) \leq 0$ and 0 otherwise).

The following theorem sheds light on the trade-off between flat versus hierarchical classification. The notion of function class capacity used here is the *empirical Rademacher complexity* (Bartlett and Mendelson, 2002; Meir and Zhang, 2003).

Theorem 1 *Let* $\mathcal{S} = ((\mathbf{x}^{(i)}, y^{(i)}))_{i=1}^m$ *be a dataset of* m *examples drawn i.i.d. according to a probability distribution* $\mathcal{D}$ *over* $\mathcal{X} \times \mathcal{Y}$*, and let* $\mathcal{A}$ *be a Lipschitz function with constant* L *dominating the* 0/1 *loss; further let* $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ *be a PSD (positive semi-definite) kernel and let* $\Phi : \mathcal{X} \rightarrow \mathbb{H}$ *be the associated feature mapping function. Assume that there exists* $R > 0$ *such that* $K(\mathbf{x}, \mathbf{x}) \leq R^2$ *for all* $\mathbf{x} \in \mathcal{X}$*. Then, for all* $1 > \delta > 0$*, with probability at least* $(1 - \delta)$ *the following hierarchical multi-class classification generalization bound holds for all* $g_f \in \mathcal{G}_{\mathcal{F}_B}$ *:*

$$\mathcal{E}(g_f) \leq \frac{1}{m} \sum_{i=1}^{m} \mathcal{A}(g_f(\mathbf{x}^{(i)}, y^{(i)})) + \frac{8BRL}{\sqrt{m}} \sum_{v \in V \setminus \mathcal{Y}} |\mathfrak{D}(v)|(|\mathfrak{D}(v)| - 1) + 3\sqrt{\frac{\ln(2/\delta)}{2m}} \quad (1)$$

where $|\mathfrak{D}(v)|$ *denotes the number of children of node* v*.*

Proof Exploiting the fact that $\mathcal{A}$ dominates the 0/1 loss and using the Rademacher data-dependent generalization bound presented in Theorem 4.9 of (Shawe-Taylor and Cristianini, 2004), one has:

$$\begin{aligned}
\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathbb{1}_{g_f(\mathbf{x},y)\leq 0} - 1\right] &\leq \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left[\mathcal{A} \circ g_f(\mathbf{x}, y) - 1\right] \\
&\leq \frac{1}{m} \sum_{i=1}^{m} (\mathcal{A}(g_f(\mathbf{x}^{(i)}, y^{(i)})) - 1) + \hat{\mathcal{R}}_m((\mathcal{A} - 1) \circ \mathcal{G}_{\mathcal{F}_B}, \mathcal{S}) + 3\sqrt{\frac{\ln(2/\delta)}{2m}}
\end{aligned}$$

where $\hat{\mathcal{R}}_m$ denotes the empirical Rademacher complexity of $(\mathcal{A}-1)\circ\mathcal{G}_{\mathcal{F}_B}$ on $\mathcal{S}$. As $x \mapsto \mathcal{A}(x)$ is a Lipschtiz function with constant L and $(\mathcal{A} - 1)(0) = 0$, we further have:

$$\hat{\mathcal{R}}_m((\mathcal{A} - 1) \circ \mathcal{G}_{\mathcal{F}_B}, \mathcal{S}) \leq 2L\hat{\mathcal{R}}_m(\mathcal{G}_{\mathcal{F}_B}, \mathcal{S})$$

with:

$$\begin{aligned}\hat{\mathcal{R}}_m(\mathcal{G}_{\mathcal{F}_B}, \mathcal{S}) &= \mathbb{E}_\sigma\left[\sup_{g_f \in \mathcal{G}_{\mathcal{F}_B}} \left|\frac{2}{m}\sum_{i=1}^{m} \sigma_i\, g_f(\mathbf{x}^{(i)}, y^{(i)})\right|\right] \\ &= \mathbb{E}_\sigma\left[\sup_{f \in \mathcal{F}_B} \left|\frac{2}{m}\sum_{i=1}^{m} \sigma_i \min_{v \in \mathfrak{P}(y^{(i)})} (f(\mathbf{x}^{(i)}, v) - \max_{v' \in \mathfrak{S}(v)} f(\mathbf{x}^{(i)}, v'))\right|\right]\end{aligned}$$

where σ_is are independent uniform random variables which take value in $\{-1, +1\}$ and are known as Rademacher variables.

Let us define the mapping c from $\mathcal{F}_B \times \mathcal{X} \times \mathcal{Y}$ into $V \times V$ as:

$$\begin{aligned}c(f, \mathbf{x}, y) = (v, v') \;\Rightarrow\; & (f(\mathbf{x}, v') = \max_{v'' \in \mathfrak{S}(v)} f(\mathbf{x}, v'')) \\ & \wedge\, (f(\mathbf{x}, v) - f(\mathbf{x}, v') = \min_{u \in \mathfrak{P}(y)} (f(\mathbf{x}, u) - \max_{u' \in \mathfrak{S}(u)} f(\mathbf{x}, u')))\end{aligned}$$

This definition is similar to the one given by Guermeur (2010) for flat multi-class classification. Then, by construction of c:

$$\hat{\mathcal{R}}_m(\mathcal{G}_{\mathcal{F}_B}, \mathcal{S}) \leq \frac{2}{m}\mathbb{E}_\sigma\left[\sup_{f \in \mathcal{F}_B} \sum_{(v,v') \in V^2, v' \in \mathfrak{S}(v)} \left|\sum_{i : c(f, \mathbf{x}^{(i)}, y^{(i)}) = (v,v')} \sigma_i (f(\mathbf{x}^{(i)}, v) - f(\mathbf{x}^{(i)}, v'))\right|\right]$$

By definition, $f(\mathbf{x}^{(i)}, v) - f(\mathbf{x}^{(i)}, v') = \langle \mathbf{w}_v - \mathbf{w}_{v'}, \Phi(\mathbf{x}^{(i)})\rangle$ and using Cauchy-Schwartz inequality:

$$\begin{aligned}\hat{\mathcal{R}}_m(\mathcal{G}_{\mathcal{F}_B}, \mathcal{S}) &\leq \frac{2}{m}\mathbb{E}_\sigma\left[\sup_{||\mathbf{W}||_{\mathbb{H}} \leq B} \sum_{(v,v') \in V^2, v' \in \mathfrak{S}(v)} \left|\langle \mathbf{w}_v - \mathbf{w}_{v'}, \sum_{i : c(f, \mathbf{x}^{(i)}, y^{(i)}) = (v,v')} \sigma_i \Phi(\mathbf{x}^{(i)})\rangle\right|\right] \\ &\leq \frac{2}{m}\mathbb{E}_\sigma\left[\sup_{||\mathbf{W}||_{\mathbb{H}} \leq B} \sum_{(v,v') \in V^2, v' \in \mathfrak{S}(v)} \|\mathbf{w}_v - \mathbf{w}_{v'}\|_{\mathbb{H}} \left\|\sum_{i : c(f, \mathbf{x}^{(i)}, y^{(i)}) = (v,v')} \sigma_i \Phi(\mathbf{x}^{(i)})\right\|_{\mathbb{H}}\right] \\ &\leq \frac{4B}{m} \sum_{(v,v') \in V^2, v' \in \mathfrak{S}(v)} \mathbb{E}_\sigma\left[\left\|\sum_{i : c(f, \mathbf{x}^{(i)}, y^{(i)}) = (v,v')} \sigma_i \Phi(\mathbf{x}^{(i)})\right\|_{\mathbb{H}}\right]\end{aligned}$$

Using Jensen's inequality, and as, $\forall i,j \in \{l|c(f,\mathbf{x}^{(l)},y^{(l)}) = (v,v')\}^2, i \neq j, \mathbb{E}_\sigma\left[\sigma_i\sigma_j\right] = 0$, we get:

$$\begin{aligned}
\hat{\mathcal{R}}_m(\mathcal{G}_{\mathcal{F}_B},\mathcal{S}) &\leq \frac{4B}{m}\sum_{(v,v')\in V^2, v'\in\mathfrak{S}(v)}\left(\mathbb{E}_\sigma\left[\left\|\sum_{i:c(f,\mathbf{x}^{(i)},y^{(i)})=(v,v')}\sigma_i\Phi(\mathbf{x}^{(i)})\right\|^2_{\mathbb{H}}\right]\right)^{1/2} \\
&= \frac{4B}{m}\sum_{(v,v')\in V^2, v'\in\mathfrak{S}(v)}\left(\sum_{i:c(f,\mathbf{x}^{(i)},y^{(i)})=(v,v')}\left\|\Phi(\mathbf{x}^{(i)})\right\|^2_{\mathbb{H}}\right)^{1/2} \\
&= \frac{4B}{m}\sum_{(v,v')\in V^2, v'\in\mathfrak{S}(v)}\left(\sum_{i:c(f,\mathbf{x}^{(i)},y^{(i)})=(v,v')}K\left(\mathbf{x}^{(i)},\mathbf{x}^{(i)}\right)\right)^{1/2} \\
&\leq \frac{4B}{m}\sum_{(v,v')\in V^2, v'\in\mathfrak{S}(v)}\left(mR^2\right)^{1/2} \\
&= \frac{4BR}{\sqrt{m}}\sum_{v\in V\setminus\mathcal{Y}}|\mathfrak{D}(v)|(|\mathfrak{D}(v)|-1)
\end{aligned}$$

Plugging this bound into the first inequality yields the desired result. □

This generalization bound proves the consistency of the ERM principle for the Pachinko-machine based method. Further, for flat multiclass classification, we recover the bounds by Guermeur (2010) by considering a hierarchy containing a root node with as many children as there are categories. Note that the definition of functions in $\mathcal{G}_{\mathcal{F}_B}$ subsumes the definition of the margin function used for the flat multiclass classification problems by Guermeur (2010), and that the factor $8L$ in the complexity term of the bound, instead of 4 by Guermeur (2010), is due to the fact that we are using an L-Lipschitz loss function dominating the 0/1 loss in the empirical Rademacher complexity. Krishnapuram et al. (2005) they provide PAC-Bayes bounds, different to ours, for Bayes Voting classifiers and Gibbs classifier under a PAC-Bayes setup (McAllester, 1998; Seeger, 2003). Lastly, Bartlett et al. (2005) have proposed tighter bounds using local Rademacher complexities. Using such bounds would lead to replace the term involving the complexity of the hierarchy in Theorem 1 by a term involving the fixed point of a sub-root function that upper bounds local Rademacher averages. Such a replacement, if it can lead to tighter bounds under some additional conditions, would however miss the explanation provided below on the behaviors of flat and hierarchical classifiers, an explanation that will be confirmed experimentally.

Flat vs hierarchical classification in large-scale taxonomies. The generalization error is controlled in inequality (1) by a trade-off between the empirical error and the Rademacher complexity of the class of classifiers. The Rademacher complexity term favors hierarchical classifiers over flat ones, as any split of a set of category of size K in p parts $K_1,\cdots,K_p$ ($\sum_{i=1}^p K_i = K$) is such that $\sum_{i=1}^p K_i^2 \leq K^2$. On the other hand, the empirical error term is likely to favor flat classifiers vs hierarchical ones, as the latter rely on a series of decisions (as many as the length of the path from the root to the chosen category in $\mathcal{Y}$) and are thus more likely to make mistakes. This fact is often referred to as the *propagation error* problem in hierarchical classification.

On the contrary, flat classifiers rely on a single decision and are not prone to this problem (even though the decision to be made is harder). When the classification problem in $\mathcal{Y}$ is highly unbalanced, then the decision that a flat classifier has to make is difficult; hierarchical classifiers still have to make several decisions, but the imbalance problem is less severe on each of them. So, in this case, even though the empirical error of hierarchical classifiers may be higher than the one of flat ones, the difference can be counterbalanced by the Rademacher complexity term, and the bound in Theorem 1 suggests that hierarchical classifiers should be preferred over flat ones.

On the other hand, when the data is well balanced, the Rademacher complexity term may not be sufficient to overcome the difference in empirical errors due to the propagation error in hierarchical classifiers; in this case, Theorem 1 suggests that flat classifiers should be preferred to hierarchical ones. These results have been empirically observed in different studies on classification in large-scale taxonomies and are further discussed in Section 6.

Similarly, one way to improve the accuracy of classifiers deployed in large-scale taxonomies is to modify the taxonomy by pruning (sets of) nodes (Wang and Lu, 2010). By doing so, one is flattening part of the taxonomy and is once again trading-off the two terms in inequality (1): pruning nodes leads to reduce the number of decisions made by the hierarchical classifier while maintaining a reasonable Rademacher complexity. Motivated from the Rademacher-based generalization error bound presented in Theorem 1, we now propose a method for pruning nodes of the given taxonomy. The output of this procedure is a new taxonomy which leads to improvement in classification accuracy when used for top-down classification.

4. Hierarchy Pruning

In this section, we present a strategy aiming at adapting the given hierarchy of classes by pruning some nodes in the hierarchy. An example of node pruning is shown in Figure 2. The rationale and motivation behind adapting the given hierarchy $\mathcal{H} = (V, E)$ to the set of input/output pair $(\mathbf{x}, y)$ is that

- Large-scale taxonomies, such as DMOZ and Yahoo! Directory, are designed with an intent of better user-experience and navigability, and not necessarily for the goal of classification,
- Taxonomy design is subject to certain degree of arbitrariness based on personal choices and preferences of the editors. Therefore, many competing taxonomies may exist, and
- The large-scale nature of such taxonomies poses difficulties in manually designing good taxonomies for classification.

The problem of pruning a hierarchy can be seen as a structure learning problem, where one wants to learn a simplified structure from a given one. The main difficulty in solving this problem is to identify the important features on which to base the decision to prune a node or not. We first present in this section a straightforward strategy that behaves well in practice but is nevertheless computationally expensive, prior to propose a "lighter" strategy based on the previous results. We will, in the next section (Section 5), introduce new theoretical results that will help us identify important features for node pruning, and

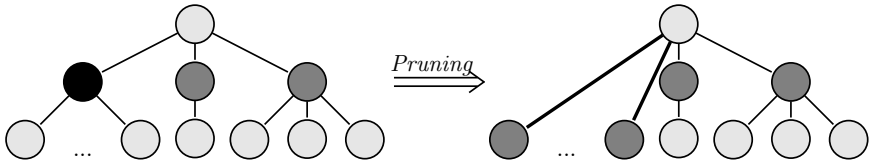

Figure 2: The pruning procedure; the node in black is replaced by its children. In the figure on the left, the gray nodes represent the siblings or sisters of node in black.

on which we will develop a greedy procedure to simplify a given hierarchy. The rationale for a greedy approach here is that optimal pruning would require evaluations of 2^k possible prunings for k siblings, which is infeasible in practice.

4.1 Hierarchy Pruning based on validation estimate

The challenge in the pruning procedure is to identify promising nodes which when pruned lead to improvement in classification accuracy. One of the simplest methods to identify such nodes is by using a validation set to check if pruning a node improves classification accuracy on that set by comparing it with accuracy obtained on the original taxonomy. This can also be interpreted as follows:

$$\text{Whether to prune a node } v? = \begin{cases} \text{Yes} & \text{If classification improves on the validation set} \\ \text{No} & \text{otherwise} \end{cases}$$

Algorithm 1 Hierarchy pruning based on validation estimate

Require: A hierarchy $\mathcal{G}$, Training set $\mathcal{S}$ and a validation set $\mathcal{S}'$
1: Train SVM classifier at each node of the tree $\mathcal{G}$ using the training set $\mathcal{S}$
2: Evaluate the accuracy of the classifier-cascade $\mathcal{G}$ on the validation set
3: **for** $v \in \mathcal{V}$ **do**
4: Prune the node v and replace it by its children
5: Re-train SVM classifier at the impacted node of the tree $\mathcal{G}'$
6: Evaluate the accuracy of the classifier-cascade $\mathcal{G}'$ on the validation set
7: **if** Cross-validation accuracy is higher on $\mathcal{G}'$ as compared to $\mathcal{G}$ **then**
8: Prune the node v
9: **else**
10: Do not prune the node v
11: **end if**
12: **end for**
13: **return** Pruned taxonomy $\mathcal{G}'$

This simple strategy is algorithmically presented in Algorithm 1. In terms of prediction accuracy, this method for pruning works reasonably well, however its main disadvantages are the computational complexity and lack of generalizability to new but somewhat related

taxonomies. In section 6, we also present experimental results obtained by this pruning strategy vis-à-vis other pruning methods presented later in this paper.

Computationally, this method requires (i) a trained cascade of top-down classifiers, (ii) for every pruned node, re-training the parent of the pruned node, and (iii) for every such node, evaluating the top-down performance on the validation set, which involves traversing the root-to-leaf path of classifier evaluation along the taxonomy. Furthermore, the steps (ii) and (iii) are to be repeated for every pruned node. Let C_{td-cas} denotes the computational complexity for training the cascade, C_v denotes the complexity for re-training of the parent node after pruning the node v, and C_{val} denotes the complexity of evaluating the validation set. Let $|V|_p$ denote the number of pruned nodes, then the complexity of the Algorithm 1 is $C_{td-cas} + |V|_p \times (C_v + C_{val})$. Due to the linear dependence on the number of pruned nodes, it becomes computationally expensive to prune a reasonably large number of nodes and check if this would result in improvement in classification accuracy of the top-down cascade. Furthermore, this process does not amount to a learning-based method for pruning and hence needs to be employed from scratch for newer taxonomies, even though these taxonomies may have similar characteristics to those encountered already.

To summarize, though quite simple, the above pruning method has the following disadvantages:

- This is a computationally expensive process to re-train the classifier at the pruned nodes and then test the performance on the validation set. As a result, this may not be applicable for large-scale taxonomies consisting of large number of internal nodes,
- This method does not amount to a learning-based strategy for pruning, and ignores data-dependent information available at the nodes of the taxonomy, and
- This process needs to be repeated for each taxonomy encountered, and information gained from taxonomy cannot be leveraged for newer taxonomies with similar characteristics.

We now turn to another method for taxonomy adaptation by pruning which is based on the generalization error analysis derived in Section 3.1. This method is computationally efficient compared to that presented in Algorithm 1 and only requires a cascade of top-down classifiers. Essentially, the criterion for pruning, which is related to the margin at each node, can be computed while training the top-down cascade. This corresponds to only the first step in Algorithm 1, and rest of the steps of evaluating on validation set are no longer required. Therefore, in terms of computational complexity, the method proposed in the next section has complexity of C_{td-cas}.

4.2 Hierarchy Pruning based on generalization-error

In this section, we present a strategy for pruning which is theoretically well motivated and is based on the generalization error bound for understanding the trade-off for flat and hierarchical classification. In view of the generalization error bound derived in Theorem 1, adapting the given taxonomy of classes aims at achieving a better trade-off between the empirical error and the error attributed to Rademacher complexity. In other words, adapting the given taxonomy $\mathcal{H}$ to the set of input output pairs $(\mathbf{x}, v)$ aims at achieving a lower value

of the bound as compared to that attained by using the original hierarchy. For a node v with parent $\pi(v)$, pruning v and replacing it by its children will increase the number of children of $\pi(v)$ and hence the associated Rademacher complexity but will decrease the empirical error along that path from root to leaf. Therefore, we need to identify those nodes in the taxonomy for which increase in the Rademacher complexity is among the lowest so that a better trade-off between the two error terms is achieved than in the original hierarchy. For this purpose, we turn to the bound on the empirical Rademacher complexity of the function class $\mathcal{G}_{\mathcal{F}_B}$.

In the derivation of Theorem 1, the empirical Rademacher complexity was upper bounded as follows:

$$\hat{\mathcal{R}}_m(\mathcal{G}_{\mathcal{F}_B}, \mathcal{S}) \leq \frac{2}{m}\mathbb{E}_\sigma \left[\sup_{||\mathbf{W}||_{\mathbb{H}} \leq B} \sum_{(v,v') \in V^2, v' \in \mathfrak{S}(v)} \|\mathbf{w}_v - \mathbf{w}_{v'}\|_{\mathbb{H}} \left\| \sum_{i:c(f,\mathbf{x}^{(i)},y^{(i)})=(v,v')} \sigma_i \Phi(\mathbf{x}^{(i)}) \right\|_{\mathbb{H}} \right] \quad (2)$$

From the above bound, we define a quantity $C(v)$ for each node v

$$C(v) = \sum_{(v,v') \in V^2, v' \in \mathfrak{S}(v)} \|\mathbf{w}_v - \mathbf{w}_{v'}\|_{\mathbb{H}} \quad (3)$$

As one can note, the right hand side of the inequality (2) above provides an upper bound on $\hat{\mathcal{R}}_m(\mathcal{G}_{\mathcal{F}_B}, \mathcal{S})$ and one can meaningfully compare only the sibling nodes since these nodes have the same training set. Thus, the explicit computation of expectation $\mathbb{E}_\sigma(.)$ with respect to the Rademacher random variables and the computation of the inner product in the feature space can be avoided. This motivates the definition of $C(v)$ in equation (3), that can be efficiently and effectively computed from the training data, and that represents a distance of node v to its sibling nodes. It must be noted that $C(v')$, for a set of siblings $\{v'\}$, as computed using equation 3 is different for each node v'.

$C(v)$ is higher when $\mathbf{w}_v$ is larger than $\mathbf{w}_{v'}$ (when measured in terms of L2-distance), for all siblings v' of v, or when $\mathbf{w}_v$ is far away from $\mathbf{w}_{v'}$ (implying that the L2-norm of the difference is large), or both. The first and last cases correspond to unbalanced classes, v being the dominant class. In such cases, pruning v by replacing it by its children leads to a more balanced problem, less prone to classification errors. Furthermore, as children of v are based on the features in v, most of them will likely be far away from the siblings of v, and the pruning, even though increasing the Rademacher complexity term, will decrease the empirical error term and, likely, the generalization error. In the second case, pruning v will lead to children that will again be, very likely, far away from the siblings of v. This pruning thus does not introduce confusion between categories and reduces the problem related to error propagations.

This suggests that an effective pruning algorithm must prune the nodes v in the taxonomy for which $C(v)$ is maximal. In practice, we focus on pruning the nodes in the top-two layers of the taxonomy. This is due to the following reasons:

- The categories in these levels represent generic concepts, such as Entertainment and Sports in Yahoo! Directory, which are typically over-lapping in nature, and
- This is also shown in the plot in Figure 3 for the average confusion of the nodes C_v^{avg} for the different levels for two of the taxonomies used in our experiments. It shows

that the confusion among the top-level nodes is much higher as compared to those in the lower levels.

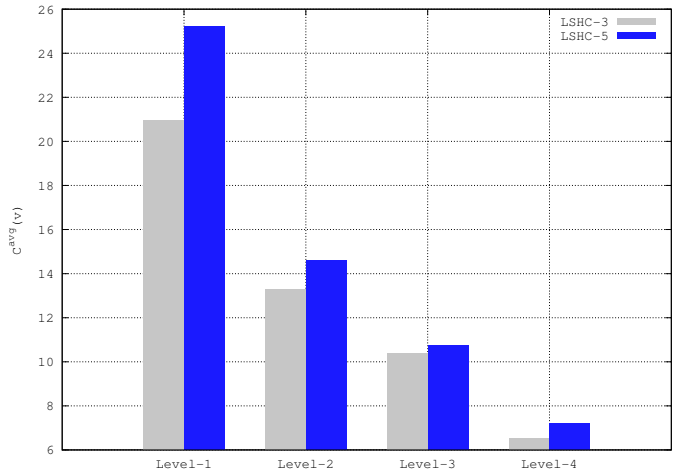

Figure 3: C_v^{avg} **plotted for various levels in the hierarchy Level 1 corresponds to the top-most level.**

Algorithm 2 The proposed method for hierarchy pruning based on Generalization Bound

Require: a hierarchy $\mathcal{G}$, Training set $\mathcal{S}$ consisting of $(\mathbf{x}, y)$ pairs, $\mathbf{x} \in \mathcal{X}$ and $y \in \mathcal{Y}$
  Train SVM classifier at each node of the tree
  $\Delta \leftarrow 0$
  for $v \in \mathcal{V}$ **do**
    Sort its child nodes $v' \in \mathfrak{D}(v)$ in decreasing order of $C(v')$
    Flatten 1st and 2nd ranked child nodes, say v'_1 and v'_2
    $\Delta = C(v'_1) - C(v'_2)$
    $v_{prev} \leftarrow v'_2$ ▷ Set the previous flattened node to v'_2
    for $v' \in \mathcal{V} - \{v'_1, v'_2\}, (v, v') \in \mathcal{E}$ **do**
      if $C(v_{prev}) - C(v') < \Delta$ **then**
        Flatten v'
        $\Delta \leftarrow C(v_{prev}) - C(v')$
        $v_{prev} \leftarrow v'$ ▷ Set the previous flattened node to v'
      else
        break
      end if
    end for
  end for
  return Pruned taxonomy $\mathcal{G}'$

The pruning process as an algorithmic procedure is shown in Algorithm 2, where the variable Δ is used to stop the pruning process in an inner iteration.

The above criterion for pruning the nodes in a large-scale taxonomy is also similar in spirit to the method introduced by Babbar et al. (2013b) which is motivated from the generalization error analysis of Perceptron Decision Trees. As shown in the experiments on large-scale datasets by using SVM and Logistic Regression classifiers, applying this strategy outputs a new taxonomy which leads to better classification accuracy as compared to the original taxonomy.

It may be noted that the pruning procedure adopts a conservative approach to avoid excessive flattening of the taxonomy. It can be modified to prune the nodes more aggressively by scaling the parameter Δ after pruning of every node, and hence allow more nodes to be pruned. However, irrespective of such design choice, this method based on the generalization bound for pruning the hierarchy has two following disadvantages :

- Higher computational complexity since one needs to learn the weight vector $\mathbf{w}_v$ for each node v in the given taxonomy. As a result, the process of identifying these nodes can be computationally expensive for large-scale taxonomies;
- It is restricted only to discriminative classifiers such as Support Vector Machines and Logistic Regression.

Therefore, we next present a meta-learning based pruning strategy for hierarchy pruning which avoids this initial training of the entire taxonomy, and is applicable to both discriminative and generative classifiers.

5. Meta-learning based pruning strategy

In this section, we present a meta-learning based generic pruning strategy which is applicable to both discriminative and generative classifiers. The meta-features for the instances are derived from the analysis of the approximation error for multi-class versions of the two well-known generative and discriminative classifiers: Naive Bayes and Logistic Regression. We then show how this generalization error analysis of the classifier at each node is combined when deployed in a typical top-down cascade of the hierarchy tree. Based on these analyses, we identify the important features that control the variation of the generalization error and determine whether a particular node should be flattened or not. We finally train a meta-classifier based on these meta-features, which predicts whether replacing a node in the hierarchy by its children (Figure 2) will improve the classification accuracy or not.

The remainder of this section is organized as follows:

1. In Section 5.1, we present asymptotic error bounds for Naive Bayes classifiers;
2. Asymptotic error bounds for Multinomial Logistic Regression classifiers are given in Section 5.2;
3. We then develop in Section 5.3:
   (a) A *pruning* bound for both types of classifiers;
   (b) A meta-classifier for pruning nodes of a taxonomy, based on features derived from both asymptotic error and pruning bounds.

Theorem 2 below is recalled by Ng and Jordan (2001). Theorems 3 and 4 provide multi-class versions of the bounds proposed by Ng and Jordan (2001) for the Naive Bayes and Logistic Regression classifiers respectively. Lastly, Theorem 5 provides a hierarchical generalization of these bounds for both classifiers. The features we are using to learn the meta-classifier are derived from Theorem 5.

5.1 Asymptotic approximation error bounds for Naive Bayes

Let us first consider a multinomial, multiclass Naive Bayes classifier in which the predicted class is the one with maximum posterior probability. The parameters of this model are estimated by maximum likelihood and we assume here that Laplace smoothing is used to avoid null probabilities. Our goal here is to derive a generalization error bound for this classifier. To do so, we recall the bound for the binomial version (directly based on the presence/absence of each feature in each document) of the Naive Bayes classifier for two target classes (Theorem 4 of (Ng and Jordan, 2001)).

Theorem 2 *For a two class classification problem in d dimensional feature space with m training examples $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^m$ sampled from distribution $\mathcal{D}$, let h and h_∞ denote the classifiers learned from the training set of finite size m and its asymptotic version respectively. Then, with high probability, the bound on misclassification error of h is given by*

$$\mathcal{E}(h) \leq \mathcal{E}(h_\infty) + G\left(O\left(\sqrt{\frac{1}{m}\log d}\right)\right) \tag{4}$$

where $G(\tau)$ represents the probability that the asymptotic classifier predicts correctly and has scores lying in the interval $(-d\tau, d\tau)$.

We extend here this result to the multinomial, multiclass Naive Bayes classifier, for a K class classification problem with $\mathcal{Y} = \{y_1, \ldots y_K\}$. To do so, we first introduce the following lemma, that parallels Lemma 3 of (Ng and Jordan, 2001):

Lemma 1 *$\forall y_k \in \mathcal{Y}$, let $\widehat{P}(y_k)$ be the estimated class probability and $P(y_k)$ its asymptotic version obtained with a training set of infinite size. Similarly, $\forall y_k \in \mathcal{Y}$ and $\forall i$, $1 \leq i \leq d$, let $\widehat{P}(w_i|y_k)$ be the estimated class conditional feature probability and $P(w_i|y_k)$ its asymptotic version (w_i denotes the i^{th} word of the vocabulary). Then, $\forall \epsilon > 0$, with probability at least $(1-\delta)$ we have :*

$$|\widehat{P}(y_k) - P(y_k)| < \epsilon,\ |\widehat{P}(w_i|y_k) - P(w_i|y_k)| < \epsilon$$

with $\delta = K\delta_0 + d\sum_{k=1}^K \delta_k$, where $\delta_0 = 2\exp(-2m\epsilon^2)$ and $\delta_k = 2d\exp(-2d_k\epsilon^2)$. d_k represents the length of class y_k, that is the sum of lengths (in number of occurrences) of all the documents in class k.

The proof of this lemma directly derives from Hoeffding's inequality and the union bound, and is a direct extension of the proof of Lemma 3 given by Ng and Jordan (2001).

Let us now denote the joint log-likelihood of the vector representation of (a document) $\boldsymbol{x}$ in class y_k by $l(\boldsymbol{x}, y_k)$:

$$l(\boldsymbol{x}, y_k) = \log\left[\widehat{P}(y_k)\prod_{i=1}^d \widehat{P}(w_i|y_k)^{x_i}\right] \tag{5}$$

where x_i represents the number of times word w_i appears in $\boldsymbol{x}$. The decision of the Naive Bayes classifier for an instance $\boldsymbol{x}$ is given by:

$$h(\boldsymbol{x}) = \underset{y_k \in \mathcal{Y}}{\operatorname{argmax}}\, l(\boldsymbol{x}, y_k) \tag{6}$$

and the one for its asymptotic version by:

$$h_\infty(\boldsymbol{x}) = \underset{y_k \in \mathcal{Y}}{\operatorname{argmax}}\, l_\infty(\boldsymbol{x}, y_k) \tag{7}$$

Lemma 1 suggests that the predicted and asymptotic log-likelihoods are close to each other, as the quantities they are based on are close to each other. Thus, provided that the asymptotic log-likehoods between the best two classes, for any given $\boldsymbol{x}$, are not too close to each other, the generalization error of the Naive Bayes classifier and the one of its asymptotic version are close to each other. Theorem 3 below states such a relationship, using the following function that measures the confusion between the best two classes for the asymptotic Naive Bayes classifier.

Definition 1 *Let $l^1_\infty(\boldsymbol{x}) = \max_{y_k \in \mathcal{Y}} l_\infty(\boldsymbol{x}, y_k)$ be the best log-likelihood score obtained for $\boldsymbol{x}$ by the asymptotic Naive Bayes classifier, and let $l^2_\infty(\boldsymbol{x}) = \max_{y_k \in \mathcal{Y} \setminus h_\infty(\boldsymbol{x})} l_\infty(\boldsymbol{x}, y_k)$ be the second best log-likelihood score for $\boldsymbol{x}$. We define the confusion of the asymptotic Naive Bayes classifier for a category set $\mathcal{Y}$ as:*

$$G_{\mathcal{Y}}(\tau) = P_{(\boldsymbol{x},y)\sim D}(|l^1_\infty(\boldsymbol{x}) - l^2_\infty(\boldsymbol{x})| < 2\tau)$$

for $\tau > 0$.

We are now in position to formulate a relationship between the generalization error of the multinomial, multiclass Naive Bayes classifier and its asymptotic version.

Theorem 3 *For a K class classification problem in d dimensional feature space with a training set of size m, $\{\boldsymbol{x}^{(i)}, y^{(i)}\}_{i=1}^m$, $\boldsymbol{x}^{(i)} \in \mathcal{X}$, $y^{(i)} \in \mathcal{Y}$, sampled from distribution $\mathcal{D}$, let h and h_∞ denote the Naive Bayes classifiers learned from a training set of finite size m and its asymptotic version respectively, and let $\mathcal{E}(h)$ and $\mathcal{E}(h_\infty)$ be their generalization errors. Then, $\forall \epsilon > 0$, one has, with probability at least $(1 - \delta_{\mathcal{Y}})$:*

$$\mathcal{E}(h) \leq \mathcal{E}(h_\infty) + G_{\mathcal{Y}}(\epsilon) \tag{8}$$

with:

$$\delta_{\mathcal{Y}} = 2K \exp\left(\frac{-2\epsilon^2 m}{C(d + d_{max})^2}\right) + 2d \exp\left(\frac{-2\epsilon^2 d_{min}}{C(d + d_{max})^2}\right)$$

where d_{max} (resp. d_{min}) represents the length (in number of occurrences) of the longest (resp. shortest) class in $\mathcal{Y}$, and C is a constant related to the longest document in $\mathcal{X}$.

Proof Using Lemma 1 and a Taylor expansion of the log function, one gets, $\forall \epsilon > 0$, $\forall \boldsymbol{x} \in \mathcal{X}$, $\forall k \in \mathcal{Y}$:

$$P\left(|l(\boldsymbol{x}, y_k) - l_\infty(\boldsymbol{x}, y_k)| < \sqrt{C}\frac{\epsilon}{\rho_0}\right) > 1 - \delta$$

where δ is the same as in Lemma 1, $\sqrt{C}$ equals to the maximum length of a document and $\rho_0 = \min_{i,k}\{P(y_k), P(w_i|y_k)\}$. The use of Laplace smoothing is important for the quantities $p(w_i|y_k)$, which may be null if word w_i is not observed in class y_k. The Laplace smoother in this case leads to $\rho_0 = \frac{1}{d+d_{max}}$. The log-likelihood functions of the multinomial, multiclass Naive Bayes classifier and the one of its asymptotic version are thus close to each other with high probability. The decision made by the trained Naive Bayes classifier and its asymptotic version on a given x only differ if the distance between the first two classes of the asymptotic classifier is less than two times the distance between the log-likelihood functions of the trained and asymptotic classifiers. Thus, using the union bound, one obtains, with probability at least $(1-\delta)$:

$$\mathcal{E}(h) \leq \mathcal{E}(h_\infty) + G_{\mathcal{Y}}\left(\epsilon\sqrt{C}(d+d_{max})\right)$$

Using a change of variable ($\epsilon' = \epsilon\sqrt{C}(d+d_{max})$) and approximating $\sum_{k=1}^{K}\exp(-2d_k\epsilon^2)$ by $\exp(-2d_{min}\epsilon^2)$, the dominating term in the sum, leads to the desired result. □

5.2 Asymptotic approximation error bounds for Multinomial Logistic Regression

We now propose an asymptotic approximation error bound for a multiclass logistic regression (`MLR`) classifier. We first consider the flat, multiclass case ($V = \mathcal{Y}$), and then show how the bounds can be combined in a typical top-down cascade, leading to the identification of important features that control the variation of these bounds.

Considering a pivot class $y^\star \in \mathcal{Y}$, a `MLR` classifier, with parameters $\boldsymbol{\beta} = \{\beta_0^y, \beta_j^y; y \in \mathcal{Y}\setminus\{y^\star\}, j \in \{1,\ldots,d\}\}$, models the class posterior probabilities via a linear function in $\mathbf{x} = (x_j)_{j=1}^d$ ((see for example Hastie et al., 2001, p. 96)) :

$$\begin{aligned} P(y|\mathbf{x};\boldsymbol{\beta})_{y\neq y^\star} &= \frac{\exp(\beta_0^y + \sum_{j=1}^d \beta_j^y x_j)}{1+\sum_{y'\in\mathcal{Y},y'\neq y^\star}\exp(\beta_0^{y'} + \sum_{j=1}^d \beta_j^{y'} x_j)} \\ P(y^\star|\mathbf{x};\boldsymbol{\beta}) &= \frac{1}{1+\sum_{y'\in\mathcal{Y},y'\neq y^\star}\exp(\beta_0^{y'} + \sum_{j=1}^d \beta_j^{y'} x_j)} \end{aligned}$$

The parameters $\boldsymbol{\beta}$ are usually fit by maximum likelihood over a training set $\mathcal{S}$ of size m (denoted by $\widehat{\boldsymbol{\beta}}_m$ in the following) and the decision rule for this classifier consists in choosing the class with the highest class posterior probability :

$$h_m(\mathbf{x}) = \operatorname*{argmax}_{y\in\mathcal{Y}} P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_m) \tag{9}$$

The following lemma states to which extent the posterior probabilities with maximum likelihood estimates $\widehat{\boldsymbol{\beta}}_m$ may deviate from their asymptotic values obtained with maximum likelihood estimates when the training size m tends to infinity (denoted by $\widehat{\boldsymbol{\beta}}_\infty$).

Lemma 2 *Let $\mathcal{S}$ be a training set of size m and let $\widehat{\boldsymbol{\beta}}_m$ be the maximum likelihood estimates of the `MLR` classifier over $\mathcal{S}$. Further, let $\widehat{\boldsymbol{\beta}}_\infty$ be the maximum likelihood estimates of parameters of `MLR` when m tends to infinity. For all examples $\mathbf{x}$, let $R > 0$ be the bound*

such that $\forall y \in \mathcal{Y}\backslash\{y^\star\}, \exp(\beta_0^y + \sum_{j=1}^d \beta_j^y x_j) < \sqrt{R}$*; then for all* $1 > \delta > 0$*, with probability at least* $(1-\delta)$ *we have:*

$$\forall y \in \mathcal{Y}, \left| P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_m) - P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_\infty) \right| < d\sqrt{\frac{R|\mathcal{Y}|\sigma_0}{\delta m}}$$

where $\sigma_0 = \max_{j,y} \sigma_j^y$ *and* $(\sigma_j^y)_{y,j}$ *represent the components of the inverse (diagonal) Fisher information matrix at* $\widehat{\boldsymbol{\beta}}_\infty$ *and are different from* σ_i *used in Section 3 wherein these represented Rademacher random variables.*

Proof By denoting the sets of parameters $\widehat{\boldsymbol{\beta}}_m = \{\hat{\beta}_j^y; j \in \{0, \ldots, d\}, y \in \mathcal{Y}\backslash\{y^\star\}\}$, and $\widehat{\boldsymbol{\beta}}_\infty = \{\beta_j^y; j \in \{0, \ldots, d\}, y \in \mathcal{Y}\backslash\{y^\star\}\}$, and using the independence assumption and the asymptotic normality of maximum likelihood estimates ((see for example Schervish, 1995, p. 421)), we have, for $0 \leq j \leq d$ and $\forall y \in \mathcal{Y} \setminus \{y^\star\}$: $\sqrt{m}(\widehat{\beta}_j^y - \beta_j^y) \sim N(0, \sigma_j^y)$ where the $(\sigma_j^y)_{y,i}$ represent the components of the inverse (diagonal) Fisher information matrix at $\widehat{\boldsymbol{\beta}}_\infty$. Let $\sigma_0 = \max_{j,y} \sigma_j^y$. Then using Chebyshev's inequality, for $0 \leq j \leq d$ and $\forall y \in \mathcal{Y}\backslash\{y^\star\}$ we have with probability at least $1 - \sigma_0/\epsilon^2$, $|\widehat{\beta}_j^y - \beta_j^y| < \frac{\epsilon}{\sqrt{m}}$. Further $\forall \mathbf{x}$ and $\forall y \in \mathcal{Y}\backslash\{y^\star\}, \exp(\beta_0^y + \sum_{j=1}^d \beta_j^y x_j) < \sqrt{R}$; using a Taylor development of the functions $\exp(x+\epsilon)$ and $(1+x+\epsilon x)^{-1}$ and the union bound, one obtains that, $\forall \epsilon > 0$ and $y \in \mathcal{Y}$ with probability at least $1 - \frac{|\mathcal{Y}|\sigma_0}{\epsilon^2}$: $\left| P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_m) - P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_\infty) \right| < d\sqrt{\frac{R}{m}}\epsilon$. Setting $\frac{|\mathcal{Y}|\sigma_0}{\epsilon^2}$ to δ, and solving for ϵ gives the result. □

Lemma 2 suggests that the predicted and asymptotic posterior probabilities are close to each other, as the quantities they are based on are close to each other. Thus, provided that the asymptotic posterior probabilities between the best two classes, for any given $\mathbf{x}$, are not too close to each other, the generalization error of the `MLR` classifier and the one of its asymptotic version should be similar. Theorem 4 below states such a relationship, using the following function that measures the confusion between the best two classes for the asymptotic `MLR` classifier defined as :

$$h_\infty(\mathbf{x}) = \operatorname*{argmax}_{y \in \mathcal{Y}} P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_\infty) \tag{10}$$

For any given $\mathbf{x} \in \mathcal{X}$, the confusion between the best two classes is defined as follows.

Definition 2 *Let* $f_\infty^1(\mathbf{x}) = \max_{y \in \mathcal{Y}} P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_\infty)$ *be the best class posterior probability for* $\mathbf{x}$ *by the asymptotic* `MLR` *classifier, and let* $f_\infty^2(\mathbf{x}) = \max_{y \in \mathcal{Y}\backslash h_\infty(\mathbf{x})} P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_\infty)$ *be the second best class posterior probability for* $\mathbf{x}$*. We define the confusion of the asymptotic* `MLR` *classifier for a category set* $\mathcal{Y}$ *as:*

$$G_{\mathcal{Y}}(\tau) = P_{(\mathbf{x},y)\sim\mathcal{D}}(|f_\infty^1(\mathbf{x}) - f_\infty^2(\mathbf{x})| < 2\tau)$$

for a given $\tau > 0$*.*

The following theorem states a relationship between the generalization error of a trained `MLR` classifier and its asymptotic version.

Theorem 4 *For a multi-class classification problem in d dimensional feature space with a training set of size m, $\{\boldsymbol{x}^{(i)}, y^{(i)}\}_{i=1}^{m}$, $\boldsymbol{x}^{(i)} \in \mathcal{X}$, $y^{(i)} \in \mathcal{Y}$, sampled i.i.d. from a probability distribution $\mathcal{D}$, let h_m and h_∞ denote the multiclass logistic regression classifiers learned from a training set of finite size m and its asymptotic version respectively, and let $\mathcal{E}(h_m)$ and $\mathcal{E}(h_\infty)$ be their generalization errors. Then, for all $1 > \delta > 0$, with probability at least $(1-\delta)$ we have:*

$$\mathcal{E}(h_m) \leq \mathcal{E}(h_\infty) + G_{\mathcal{Y}}\left(d\sqrt{\frac{R|\mathcal{Y}|\sigma_0}{\delta m}}\right) \tag{11}$$

where $\sqrt{R}$ is a bound on the function $\exp(\beta_0^y + \sum_{j=1}^{d}\beta_j^y x_j)$, $\forall \boldsymbol{x} \in \mathcal{X}$ and $\forall y \in \mathcal{Y}$, and σ_0 is a constant.

Proof The difference $\mathcal{E}(h_m) - \mathcal{E}(h_\infty)$ is bounded by the probability that the asymptotic `MLR` classifier h_∞ correctly classifies an example $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$ randomly chosen from $\mathcal{D}$, while h_m misclassifies it. Using Lemma 2, for all $\delta \in (0,1), \forall \mathbf{x} \in \mathcal{X}, \forall y \in \mathcal{Y}$, with probability at least $1-\delta$, we have:

$$\left|P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_m) - P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_\infty)\right| < d\sqrt{\frac{R|\mathcal{Y}|\sigma_0}{\delta m}}$$

Thus, the decision made by the trained MLR and its asymptotic version on an example $(\mathbf{x}, y)$ differs only if the distance between the two predicted classes of the asymptotic classifier is less than two times the distance between the posterior probabilities obtained with $\widehat{\boldsymbol{\beta}}_m$ and $\widehat{\boldsymbol{\beta}}_\infty$ on that example; and the probability of this is exactly $G_{\mathcal{Y}}\left(d\sqrt{\frac{R|\mathcal{Y}|\sigma_0}{\delta m}}\right)$, which upper-bounds $\mathcal{E}(h_m) - \mathcal{E}(h_\infty)$. □

Note that the quantity σ_0 in Theorem 4 represents the largest value of the inverse (diagonal) Fisher information matrix ((Schervish, 1995)), and thus corresponds to the inverse of the smallest value of the (diagonal) Fisher information matrix, which corresponds to the smallest amount of information one has on the estimation of each parameter $\widehat{\beta}_j^k$. This smallest amount of information is in turn related to the length (in number of occurrences) of the longest (resp. shortest) class in $\mathcal{Y}$ denoted respectively by d_{max} and d_{min} as, the smaller they are, the larger σ_0 is likely to be.

5.3 A learning based node pruning strategy

Let us now consider a hierarchy of classes and a top-down classifier making decisions at each level of the hierarchy. A node-based pruning strategy can be easily derived from the approximation bounds above. Indeed, any node v in the hierarchy $\mathcal{H} = (V, E)$ is associated with three category sets: its sibling categories with the node itself $\mathfrak{S}'(v) = \mathfrak{S}(v) \cup \{v\}$, its children categories, $\mathfrak{D}(v)$, and the union of its siblings and children categories, denoted $\mathfrak{F}(v) = \mathfrak{S}(v) \cup \mathfrak{D}(v)$.

These three sets of categories are the ones involved before and after the pruning of node v. Let us now denote by $h_m^{\mathfrak{S}'_v}$ a classifier learned from a set of sibling categories of node v and the node itself, and by $h_m^{\mathfrak{D}_v}$ a classifier learned from the set of children categories of node v ($h_\infty^{\mathfrak{S}'_v}$ and $h_\infty^{\mathfrak{D}_v}$ respectively denote their asymptotic versions). The following theorem is a direct extension of Theorems 3 and 4 to this setting.

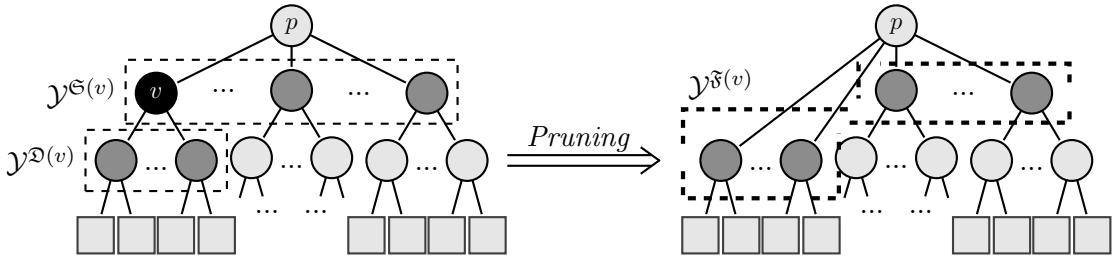

Figure 4: The pruning procedure for a candidate class node u (in black). After replacing the candidate node by its children, the new category set $\mathcal{Y}^{\mathfrak{F}(v)}$ contains the classes from both the daughter and the sister category sets of v.

Theorem 5 *Using notations from both Theorems 3 and 4, $\forall \epsilon > 0$, $v \in V \setminus \mathcal{Y}$, one has:*

$$\mathcal{E}(h_m^{\mathfrak{S}'_v}) + \mathcal{E}(h_m^{\mathfrak{D}_v}) \leq \mathcal{E}(h_\infty^{\mathfrak{S}'_v}) + \mathcal{E}(h_\infty^{\mathfrak{D}_v}) + G_{\mathfrak{S}'(v)}(\epsilon) + G_{\mathfrak{D}(v)}(\epsilon)$$

with probability at least $1 - \left(\frac{Rd^2 |\mathfrak{S}'(v)| \sigma_0^{\mathfrak{S}'(v)}}{m_{\mathfrak{S}'(v)} \epsilon^2} + \frac{Rd^2 |\mathfrak{D}(v)| \sigma_0^{\mathfrak{D}(v)}}{m_{\mathfrak{D}(v)} \epsilon^2} \right)$ *for* `MLR` *classifiers and with probability at least* $1 - \left(\delta_{\mathfrak{S}'(v)} + \delta_{\mathfrak{D}(v)}\right)$ *for Naive Bayes classifiers, with $\delta_{\mathcal{Y}}$ defined in Theorem 3.*

$\{|\mathcal{Y}^\ell|, m_{\mathcal{Y}^\ell}, \sigma_0^{\mathcal{Y}^\ell}, d_{max}^\ell, d_{min}^\ell; \mathcal{Y}^\ell \in \{\mathfrak{S}'(v), \mathfrak{D}(v)\};\}$ are constants related to the set of categories $\mathcal{Y}^\ell \in \{\mathfrak{S}'(v), \mathfrak{D}(v)\}$ and involved in the respective bounds stated in Theorem 3 and 4. Denoting by $h_m^{\mathfrak{F}_v}$ the classifier trained on the set $\mathfrak{F}(v)$ and by $h_\infty^{\mathfrak{F}_v}$ its asymptotic version, Theorem 5 suggests that one should prune node v if:

$$G_{\mathfrak{F}(v)}(\epsilon) \leq G_{\mathfrak{S}'(v)}(\epsilon) + G_{\mathfrak{D}(v)}(\epsilon) \tag{12a}$$

and, for `MLR` classifiers:

$$\frac{|\mathfrak{F}(v)| \sigma_0^{\mathfrak{F}(v)}}{m_{\mathfrak{F}(v)}} \leq \frac{|\mathfrak{S}'(v)| \sigma_0^{\mathfrak{S}'(v)}}{m_{\mathfrak{S}'(v)}} + \frac{|\mathfrak{D}(v)| \sigma_0^{\mathfrak{D}(v)}}{m_{\mathfrak{D}(v)}} \tag{12b}$$

or, for Naive Bayes classifiers:

$$\delta_{\mathfrak{F}(v)} \leq \delta_{\mathfrak{S}'(v)} + \delta_{\mathfrak{D}(v)} \tag{12c}$$

The above conditions for pruning a node v rely on the union bound and thus are not likely to be exploitable in practice. They nevertheless exhibit the factors that play an important role in assessing whether a particular trained classifier is close or not to its asymptotic version. Following Definitions 1 and 2, $G_{\mathcal{Y}}(\epsilon)$ is of the form:

$$G_{\mathcal{Y}}(\epsilon) = P_{(\mathbf{x},y)\sim\mathcal{D}}(|g_\infty(\mathbf{x}) - g_\infty(\mathbf{x})| < 2\epsilon)$$

where g corresponds to the best log-likelihood for Naive Bayes classifiers or the best class posterior for `MLR` classifiers. As it relies on the unknown distribution $\mathcal{D}$ and asymptotic log-likelihood $g_\infty()$, $G_{\mathcal{Y}}(.)$ cannot be computed directly. It however measures the confusion

Features for node $v \in V$	
1. # of classes in level ℓ, K^ℓ	2. Vocabulary size, d^ℓ
3. # of docs m^ℓ	4. d^ℓ_{min}
5. d^ℓ_{max}	6. $\frac{m^\ell}{(d^\ell+d^\ell_{max})^2}$
7. $\frac{d^\ell_{min}}{(d^\ell+d^\ell_{max})^2}$	8. $cos_{inter}(\ell)$
9. $\mathrm{KL}_{inter}(\ell)$	

Table 1: Features involved in the vector representation of a node $v \in \mathcal{H} = (V, E)$. As $\ell \in \{\mathfrak{F}(v), \mathfrak{D}(v), \mathfrak{S}(v)\}$, we have in total 27 features associated with the different category sets considered for flattening node u.

between categories and can thus be approximated by measures of the similarity between classes. We propose here to estimate this confusion with two simple quantities: the average cosine similarity of all the pairs of classes in $\mathcal{Y}$, and the average symmetric Kullback-Leibler divergences between all the pairs in $\mathcal{Y}$ according to class conditional multinomial distributions. Each node $v \in V$ when deployed with `MLR` and Naive Bayes classifiers can then be characterized by factors shown in Table 1, which are involved in the estimation of inequality (12) above.

The average cosine similarity and the average symmetric KL divergence are calculated as follows for each pair of nodes (u, v) in level l:

$$cos_{inter}(\ell) = \frac{1}{\sum_{k=1}^{K^\ell} \sum_{k'=1 k' \neq k}^{K^\ell} m_{u_k^\ell} m_{v_{k'}^\ell}} \sum_{k=1}^{K^\ell} \sum_{k'=1 k' \neq k}^{K^\ell} m_{u_k^\ell} m_{v_{k'}^\ell} cos(u_k^\ell, v_{k'}^\ell)$$

$$KL_{inter}(\ell) = \frac{1}{K^\ell(K^\ell - 1)} \sum_{k=1}^{K^\ell} \sum_{k'=1 k' \neq k}^{K^\ell} KL(Q_{u_k^\ell} || Q_{v_{k'}^\ell}) + KL(Q_{v_{k'}^\ell} || Q_{u_k^\ell}),$$

where $KL(Q_u||Q_v)$ denotes the Kullback-Leibler divergence between the class conditional probability distributions of the features present in the two classes u and v:

$$KL(Q_u||Q_v) = \sum_{w \in u,v} p(w|u) \log \frac{p(w|u)}{p(w|v)}$$

where $p(w|u)$ denotes the probability of word/feature w in class u, taking smoothing into account.

Algorithm 3 presents the process of learning the hierarchy pruning by learning a meta-classifier from the meta-features as mentioned above. The procedure for collecting training data associates a positive (resp. negative) class to a node if the pruning of that node leads to a final performance increase (resp. decrease). A meta-classifier is then trained on these features using a training set from a selected class hierarchy. After the learning phase, the meta-classifier is applied to each node of a new hierarchy of classes so as to identify which

Algorithm 3 The pruning strategy.

```
1: procedure PRUNE HIERARCHY(a hierarchy H, a meta-classifier C_m)
2:     clist[] ← H.root;                                  ▷ Initialize with root node
3:     while !clist.isEmpty() do
4:         parent ← clist.getNext()
5:         list[] ← Ch(parent);                           ▷ Candidate children to merge
6:         while !list.isEmpty() do
7:             index ← MERGE(parent,list,C_m);            ▷ Index of children to be merged
8:             if index == −1 then
9:                 break;
10:            end if
11:            list.add(Ch(list[index]))                  ▷ Move up the children of node list[index]
12:            list.remove(index);                        ▷ This node has been merged
13:        end while
14:        clist.add(Ch(clist[j]));                       ▷ Adds next level parents
15:    end while
16:    export new hierarchy;
17: end procedure
```

nodes should be pruned. A simple strategy to adopt is then to prune nodes in sequence: starting from the root node, the algorithm checks which children of a given node v should be pruned (lines 6-13) by creating the corresponding meta-instance and feeding the meta-classifier; the child that maximizes the probability of the positive class is then pruned; as the set of categories has changed, we recalculate which children of v can be pruned, prune the best one (as above) and iterate this process till no more children of v can be pruned; we then proceed to the children of v and repeat the process. The function MERGE takes as arguments the parent, the candidate children to be merged as well as the meta-classifier. It returns the index of the child that should be merged. In case where the meta-classifier does not identify any child eligible for merging then the pruning procedure continues with the next parent in the list. The meta-classifier tries to predict whether pruning a specific node will lead to an increase of the performance over 10%. That means that the classifier may identify several candidate nodes and one of them will be selected for pruning.

6. Experimental Analysis

We start our discussion by presenting results on different hierarchical datasets with different characteristics using `MLR` and `SVM` classifiers. The datasets we used in these experiments are two large datasets extracted from the International Patent Classification (**IPC**) dataset[3] and the publicly available DMOZ dataset from the second LSHTC challenge (**LSHTC2**)[4]. Both datasets are multi-class; **IPC** is single-label and **LSHTC2** multi-label with an average of 1.02 categories per class. We created 5 datasets from **LSHTC2** by splitting randomly the first layer nodes (11 in total) of the original hierarchy in disjoint subsets. The classes

3. `http://www.wipo.int/classifications/ipc/en/support/`
4. `http://lshtc.iit.demokritos.gr/`

Dataset	# Tr.	# Test	# Classes	# Feat.	Depth	CR	Error ratio
LSHTC2-1	25,310	6,441	1,789	145,859	6	0.008	1.24
LSHTC2-2	50,558	13,057	4,787	271,557	6	0.003	1.32
LSHTC2-3	38,725	10,102	3,956	145,354	6	0.004	2.65
LSHTC2-4	27,924	7,026	2,544	123,953	6	0.005	1.8
LSHTC2-5	68,367	17,561	7,212	192,259	6	0.002	2.12
IPC	46,324	28,926	451	1,123,497	4	0.02	12.27

Table 2: Datasets used in our experiments along with the properties: number of training examples, test examples, classes and the size of the feature space, the depth of the hierarchy and the complexity ratio of hierarchical over the flat case ($\sum_{v\in V\setminus\mathcal{Y}} |\mathfrak{D}(v)|(|\mathfrak{D}(v)|-1)/|\mathcal{Y}|(|\mathcal{Y}|-1)$), the ratio of empirical error for hierarchical and flat models.

for the **IPC** and **LSHTC2** datasets are organized in a hierarchy in which the documents are assigned to the leaf categories only. Table 2 presents the characteristics of the datasets.

CR denotes the complexity ratio between hierarchical and flat classification, given by the Rademacher complexity term in Theorem 1: $\left(\sum_{v\in V\setminus\mathcal{Y}} |\mathfrak{D}(v)|(|\mathfrak{D}(v)|-1)\right) / (|\mathcal{Y}|(|\mathcal{Y}|-1))$; the same constants B, R and L are used in the two cases. As one can note, this complexity ratio always goes in favor of the hierarchical strategy, although it is 2 to 10 times higher on the **IPC** dataset, compared to **LSHTC2-1,2,3,4,5**. On the other hand, the ratio of empirical errors (last column of Table 2) obtained with top-down hierarchical classification over flat classification when using `SVM` with a linear kernel is this time higher than 1, suggesting the opposite conclusion. The error ratio is furthermore really important on **IPC** compared to **LSHTC2-1,2,3,4,5**. The comparison of the complexity and error ratios on all the datasets thus suggests that the flat classification strategy may be preferred on **IPC**, whereas the hierarchical one is more likely to be efficient on the **LSHTC** datasets. This is indeed the case, as is shown below.

To test our simple node pruning strategy, we learned binary classifiers aiming at deciding whether to prune a node, based on the node features described in the previous section. The label associated to each node in this training set is defined as +1 if pruning the node increases the accuracy of the hierarchical classifier by at least 0.1, and -1 if pruning the node decreases the accuracy by more than 0.1. The threshold at 0.1 is used to avoid too much noise in the training set. The meta-classifier is then trained to learn a mapping from the vector representation of a node (based on the above features) and the labels $\{+1; -1\}$. We used the first two datasets of **LSHTC2** to extract the training data while **LSHTC2-3**, **4**, **5** and **IPC** were employed for testing.

The procedure for collecting training data is repeated for the `MNB`, `MLR` and `SVM` classifiers resulting in three meta-datasets of 119 (19 positive and 100 negative), 89 (34 positive and 55 negative) and 94 (32 positive and 62 negative) examples respectively. For the binary classifiers, we used AdaBoost with random forest as a base classifier, setting the number of trees to 20, 50 and 50 for the `MNB`, `MLR` and `SVM` classifiers respectively and leaving the other parameters at their default values. Several values have been tested for the number

of trees ({10, 20, 50, 100 and 200}), the depth of the trees ({unrestricted, 5, 10, 15, 30, 60}), as well as the number of iterations in AdaBoost ({10, 20, 30}). The final values were selected by cross-validation on the training set (**LSHTC2-1** and **LSHTC2-2**) as the ones that maximized accuracy and minimized false-positive rate in order to prevent degradation of accuracy. For example, Table 6 presents the true positive and false positive rates for the two classes for the meta-dataset of `MLR`.

Class	TP rate	FP rate
Prune (+1)	0.737	0.020
Do not prune (-1)	0.980	0.263

Table 3: True positive and false positive rates for the `MLR` meta-daset (119 examples).

We consider three different classifiers which include Multinomial Naive Bayes (`MNB`), Multi-class Logistic Regression (`MLR`) and Support Vector Machine (`SVM`) classifiers. The configurations of the taxonomy that we consider are fully flat classifier (FL), fully hierarchical (FH) top-down *Pachinko* machine, a random pruning (RN), and the two proposed pruning methods which include (i) Bound-based pruning strategy (PR-B) given in Section 4 and (ii) Meta-learning based pruning strategy (PR-M) proposed in Algorithm 3. For the PR-M pruning method, our experimental setup involves two scenarios: (i) The meta-classifier is trained over one kind of DMOZ hierarchy (LSHTC2-1, and LSHTC-2) and tested over another DMOZ hierarchy (LSHTC-3, LSHTC-4 and LSHTC-5), such that both sets have similar characteristics, and secondly, (ii) The meta-classifier is trained over one set of DMOZ hierarchy (LSHTC2-1, and LSHTC-2) and tested over IPC hierarchy which is derived from a different domain (patent classification) such that both sets have much less common characteristics. In this sense, our meta-classifier could learn to learn over one doamin and apply the learnt knowledge to another domain. For the random pruning we restrict the procedure to the first two levels and perform randomly prune 4 nodes (this is the average number of nodes that are pruned in the PR-M and PR-B strategies). We also present results for another pruning strategy proposed in our earlier work (Babbar et al., 2013b) which is based on error analysis of Perceptron Decision Trees (Bennett et al., 2000), and is referred to as PR-P. The results for the naive pruning method based on estimate on a validation set (as described in Section 4.1) are also presented, and referred to as PR-V. For each dataset we perform 5 independent runs for the random pruning and we record the best performance. For `MLR` and `SVM`, we use the LibLinear library (Fan et al., 2008) and use squared hinge-loss with $L2$-regularized versions, setting the penalty parameter C by cross-validation.

6.1 Flat versus Hierarchical classification

The classification error results on the test set of **LSHTC2-3,4,5** and **IPC** are reported in Table 4. On all **LSHTC** datasets flat classification performs worse than the fully hierarchy top-down classification, for all classifiers. These results are in line with complexity and empirical error ratios for `SVM` estimated on different collections and shown in table 2 as well as with the results obtained in Liu et al. (2005); Dumais and Chen (2000) over the same type

	LSHTC2-3			LSHTC2-4			LSHTC2-5			IPC		
	MNB	MLR	SVM	MNB	MLR	SVM	MNB	MLR	SVM	MNB	MLR	SVM
FL	73.0$^{\downarrow\downarrow}$	52.8$^{\downarrow\downarrow}$	53.5$^{\downarrow\downarrow}$	84.9$^{\downarrow\downarrow}$	49.7$^{\downarrow\downarrow}$	50.1$^{\downarrow\downarrow}$	83.9$^{\downarrow\downarrow}$	54.2$^{\downarrow\downarrow}$	54.7$^{\downarrow\downarrow}$	67.2$^{\downarrow\downarrow}$	54.6	**46.6**
RN	61.9$^{\downarrow\downarrow}$	49.3$^{\downarrow\downarrow}$	51.7$^{\downarrow\downarrow}$	70.5$^{\downarrow\downarrow}$	47.8$^{\downarrow\downarrow}$	48.4$^{\downarrow\downarrow}$	69.0$^{\downarrow\downarrow}$	53.2$^{\downarrow\downarrow}$	53.6$^{\downarrow}$	64.3$^{\downarrow\downarrow}$	54.7$^{\downarrow}$	50.2$^{\downarrow\downarrow}$
FH	62.0$^{\downarrow\downarrow}$	48.4$^{\downarrow\downarrow}$	49.8$^{\downarrow\downarrow}$	68.3$^{\downarrow}$	47.3$^{\downarrow\downarrow}$	47.6$^{\downarrow}$	65.6$^{\downarrow}$	52.6$^{\downarrow}$	52.7	64.4$^{\downarrow}$	55.2$^{\downarrow}$	51.3$^{\downarrow\downarrow}$
PR-V	61.7	48.2	49.5	65.9	46.7	46.6	67.7	52.3	52.2	63.7	54.6	50.3
PR-B	-	48.1	49.5	-	**46.6**	**46.5**	-	**52.2**	**52.2**	-	54.5	50.5
PR-M	61.3	**48.0**	**49.3**	65.4	46.9	47.2	67.8	**52.2**	52.3	63.9	**54.4**	50.7
PR-P	-	48.3	49.6	-	**46.6**	46.8	-	52.4	52.3	-	54.5	50.9

Table 4: Error results across all datasets. Bold typeface is used for the best results. Statistical significance (using micro sign test (s-test) as proposed in (Yang and Liu, 1999)) is denoted with $^{\downarrow}$ for p-value<0.05 and with $^{\downarrow\downarrow}$ for p-value<0.01.

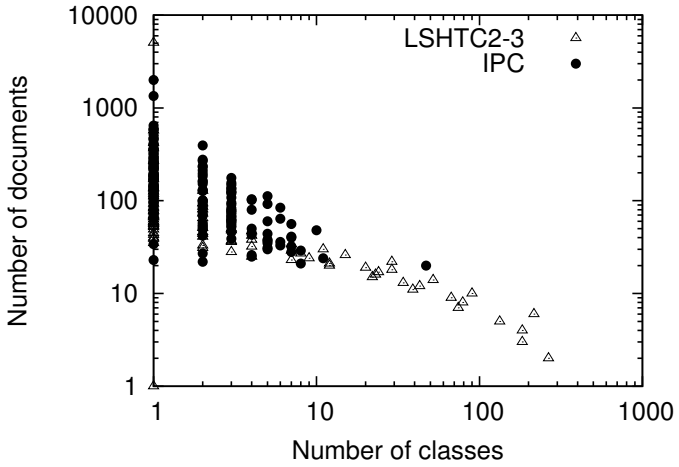

Figure 5: Number of classes (on X-axis) which have the specified number of documents (on Y-axis) for **LSHTC2-3** dataset and **IPC** dataset

of taxonomies. Further, the work by Liu et al. (2005) demonstrated that class hierarchies on **LSHTC** datasets suffer from *rare categories* problem, i.e., 80% of the target categories in such hierarchies have less than 5 documents assigned to them. The value for Macro-F1 measure which weighs all leaf-level classes equally (in contrast to Micro-F1 which weighs each example in the test set equally) is given in Table 5. Macro-F1 measure is particularly interesting when dealing with datasets consisting of rare-categories, which is typically the case in most naturally occurring category systems such as DMOZ and Wikipedia.

As a result, flat methods on such datasets face unbalanced classification problems which results in smaller error ratios; hierarchical classification should be preferred in this case. On the other hand, for hierarchies such as the one of **IPC**, which are relatively well balanced

	LSHTC2-3			LSHTC2-4			LSHTC2-5			IPC		
	MNB	MLR	SVM	MNB	MLR	SVM	MNB	MLR	SVM	MNB	MLR	SVM
FL	17.1$^{\downarrow\downarrow}$	31.1$^{\downarrow\downarrow}$	31.6 $^{\downarrow\downarrow}$	15.1$^{\downarrow\downarrow}$	33.1 $^{\downarrow\downarrow}$	32.9$^{\downarrow\downarrow}$	15.0$^{\downarrow\downarrow}$	29.2$^{\downarrow\downarrow}$	29.1$^{\downarrow\downarrow}$	25.8 $^{\downarrow\downarrow}$	47.9	**51.2**
RN	20.2$^{\downarrow\downarrow}$	32.2$^{\downarrow\downarrow}$	31.9$^{\downarrow}$	19.2$^{\downarrow}$	33.6 $^{\downarrow}$	33.2$^{\downarrow\downarrow}$	18.1$^{\downarrow}$	29.9$^{\downarrow\downarrow}$	29.9$^{\downarrow\downarrow}$	26.1 $^{\downarrow}$	45.2$^{\downarrow\downarrow}$	47.8$^{\downarrow\downarrow}$
FH	22.1$^{\downarrow}$	32.8$^{\downarrow}$	32.2	20.1$^{\downarrow}$	34.1$^{\downarrow}$	33.7$^{\downarrow}$	18.9$^{\downarrow}$	30.5$^{\downarrow}$	30.7	26.2 $^{\downarrow}$	44.2$^{\downarrow}$	46.5$^{\downarrow}$
PR-V	22.3	33.0	32.1	21.2	34.6	34.3	19.4	31.7	31.7	26.4	48.1	48.1
PR-B	-	33.1	32.3	-	34.7	**34.4**	-	**31.8**	**31.9**	-	48.1	47.9
PR-M	22.4	**33.2**	**32.4**	21.2	**34.8**	34.3	19.3	31.7	31.8	26.5	**48.2**	48.7
PR-P	-	33.0	32.2	-	34.4	34.3	-	31.6	31.5	-	48.0	47.6

Table 5: Macro-F1 results across all datasets. Bold typeface is used for the best results. Statistical significance (using macro-level t-test as proposed in (Yang and Liu, 1999)) is denoted with $^{\downarrow}$ for p-value<0.05 and with $^{\downarrow\downarrow}$ for p-value<0.01.

and do not suffer from the rare categories phenomenon, flat classification performs at par or even better than hierarchical classification. The difference in the distribution of data among leaf-level categories for the **LSHTC** datasets and **IPC** dataset is illustrated in Figure 5 on log-log scale. As one can note, in most categories **IPC** have a lot (from tens to few hundreds) of documents which belong to them as denoted by the triangles. On the other hand, **LSHTC2-3** dataset has a lot of classes with a small number (1 or 2) of documents as shown by the high concentration of solid dots near the Y-axis. In this respect, the fit to power-law distribution of documents among categories in large-scale taxonomies has also been studied recently (Babbar et al., 2014a). The relative performance between the flat and top-down approaches on the two kinds of datasets is in agreement with the conclusions obtained in recent studies in which the datasets considered do not have *rare categories* and are more well-balanced (Bengio et al., 2010; Gao and Koller, 2011; Perronnin et al., 2012; Deng et al., 2011).

We would also like to mention that using the top-down classification for mono-label prediction, we obtained an accuracy of 36.6% on the original LSHTC2 dataset. On the other hand, the top approach was a neural network based approach which used the echo-state network, and achieved an accuracy of 38.8%. Unlike the participants in LSHTC whose goal was to maximize the accuracy by using techniques such as ensemble methods and feature engineering, our goal in this work is to theoretical analyze flat versus hierarchical classification and extend the framework for performing taxonomy adaptation. Furthermore, the original LSHTC2 dataset was mildly multi-label, and since our top-down prediction algorithm predicts only a single label, we would loose some performance as compared to algorithms which are aimed towards making multi-label predictions.

6.2 Effect of pruning

The proposed hierarchy pruning strategies aim to adapt the given taxonomy structure for better classification while maintaining the ancestor-descendant relationship between a given pair of nodes. We compare the four pruning strategies, (i) first, based on estimation

on a validation set (PR-V) as given in Section 4.1, (ii) second, based on minimizing the Rademacher-based generalization error bound (PR-B) as given in Section 4.2, (iii) third, based on meta-learning (PR-M) as described in Section 5, and (iv) fourth, based on a method presented by Babbar et al. (2013b) (PR-P) against the random pruning (RN) and fully hierarchical (FH) classification. As shown in Table 4, the proposed pruning strategies lead to statistically significant better results for all three classifiers compared to both the original taxonomy and a randomly pruned one. Since Rademacher-bound based pruning strategy, PR-B, is similar to PR-P proposed by Babbar et al. (2013b), they have almost similar performance on most datasets. A similar result is reported by Wang and Lu (2010) through a pruning of an entire layer of the hierarchy, which can be seen as a generalization, even though empirical in nature, of the pruning strategy retained here. Another interesting approach to modify the original taxonomy is presented by Zhang et al. (2006). In this study, three other elementary modification operations are considered, again with an increase of performance. It is also worth noticing that the pruning strategy based on estimation on a validation set also leads to improvement in classification accuracy. However, as discussed earlier, the method is computationally expensive and unlike the meta-learning based pruning method, this does not amount to a learning algorithm which can be applied to unseen but similar taxonomies.

For `MNB` classifier, one can notice that the proposed pruning method (PR-M) based on meta-learning has the best performance in all datasets achieving significantly better results compared to its rivals. This shows that flattening the hierarchy can boost the performance, even in situations where the fully hierarchical classifier is better than its flat version (this is the case for all the datasets considered for `MNB`). The random pruning achieves slightly better accuracies than FH in **LSHTC2-3** and **IPC** datasets, but is in general in between the performance of the flat classifier and its fully hierarchical version. Statistical significance tests report significant differences in favor of the proposed approach for pruning. We also observe that all hierarchical methods consistently outperform the flat case. This is an expected result as the flat `MNB` classifier suffers from the problem of unbalanced data. The difference between the performance of the flat `MNB` classifier and its hierarchical versions is less marked for the **IPC** dataset.

For `MLR` and `SVM` classifiers, both pruning approaches have better performance in all datasets compared to its rivals, the difference being significant in all cases but with the flat classifier on **IPC**. One can also notice that due to the balanced nature of the **IPC** dataset, the performance of the flat classifier is close to that of hierarchical methods. For the same reason, random pruning is also more effective in the **IPC** dataset as compared to other datasets. Comparing the respective behaviors of the `MLR` and `SVM` against `MNB`, one can note that `MLR` and `SVM` are more robust to variations in the taxonomy as compared to `MNB`. This is reflected in much lesser variation in the accuracy for these classifiers under different configurations of the hierarchy. Lastly, and not surprisingly, the performance of `MLR` and `SVM` are much better than that of `MNB` on all the datasets considered here.

Here we would like to stress that we follow a greedy strategy in both the pruning methods (PR-B and PR-M) which rank the sibling nodes in the order these should be pruned. In case of PR-B, this is measured using the confusion $C(v)$ for each node and for PR-M, this is given by the confidence of the meta-classifier which return this as a probability estimate. This is due to the fact that picking up the optimal subset of nodes to prune from

all possible subsets of children nodes (say K') would require considering all possible $2^{K'}$ subsets. Furthermore, considering all possible orders to pick the nodes one-by-one would require considering all possible permutations of ordering. Both of these have exponential computational complexity in the number of children nodes.

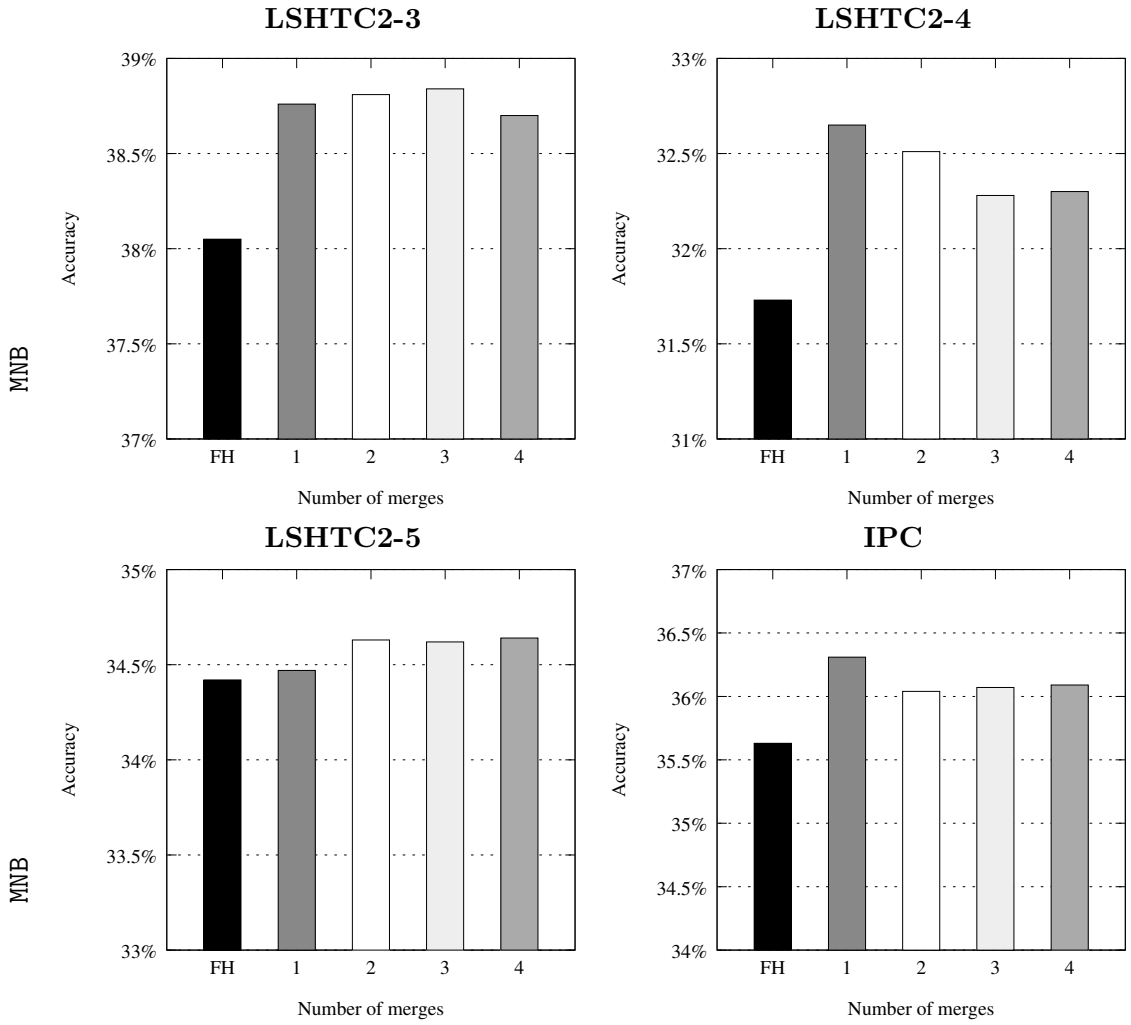

Figure 6: Accuracy performance with respect to the number of pruned nodes for `MNB` on different test sets.

6.3 Effect of number of pruned nodes for meta-learning based pruning strategy

For studying how the performance changes according to the number of pruned nodes, we record the accuracy of the proposed pruning method for 1 to 4 number of prunings. Note that pruning of nodes is done in sequence and is not independent. The results for both

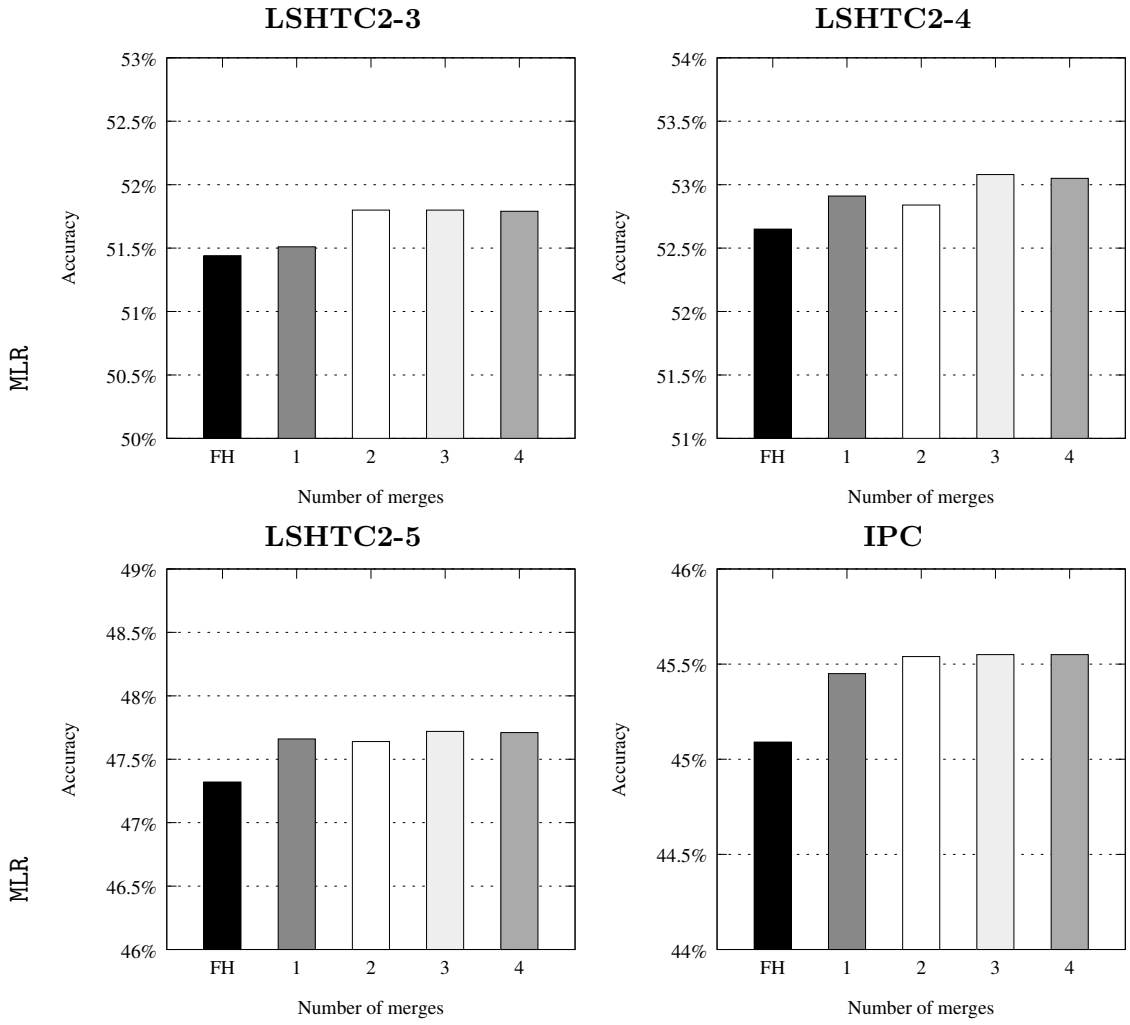

Figure 7: Accuracy performance with respect to the number of pruned nodes for `MLR` (down) on different test sets.

`MNB` and `MLR` are depicted in Figures 6 and 7, with a comparison to the FH method. The comparison with `SVM` is not explicitly shown as its behavior is similar to `MLR` classifier.

Interestingly, across all datasets, the proposed method has better performance than FH for all number of pruning nodes for both `MNB` and `MLR`. This shows that the proposed method is able to select appropriate nodes in the hierarchy for pruning. Additionally, we note that in the majority of cases the first pruned node provides a higher increase in accuracy than the following nodes. This is an expected behavior as the first prunings are typically performed at the upper level and thus tend to have a higher impact (as they will be used in more in the classification of more documents) than the nodes pruned done at lower levels. We want to stress here the fact that the performance with respect to the number of pruned nodes is affected by several factors, as the accuracy of the meta-classifier, the level of the hierarchy

Dataset	Logistic			Hinge		
	PR-M	LOMTree	FilterTree	PR-M	LOMTree	FilterTree
LSHTC2-3	48.1	60.6	60.3	49.3	60.2	59.8
LSHTC2-4	46.9	72.7	73.6	47.2	72.7	72.6
LSHTC2-5	52.2	72.0	72.7	52.3	70.5	71.0
IPC	54.4	73.9	65.1	50.7	68.8	67.1

Table 6: Comparison of classification errors for the proposed pruning method (PR-M) with approaches that build the hierarchical structure.

where the nodes are pruned and their sequence. For example, in dataset **LSHTC2-4** (Figure 6), there is a drop of performance after the first flattening which we believe is due to false positives provided by the meta-classifier. As shown for `MLR` in Figure 7 and across all datasets that the behavior of the pruning method is more stable without decrease in the final performance.

6.4 Building Taxonomy

We also compare the proposed method with approaches that construct a hierarchy with logarithmic depth over the labels of the problem. Such methods are extremely useful in cases where there is no hierarchical structured information available. More specifically, we compare with LOMTree (Choromanska and Langford, 2014) and FilterTree (Beygelzimer et al., 2009b), both implemented in the Vowpal Wabbit open source system.[5] LOMTree creates binary trees assigning each time the examples to the two respective children of a parent node. When the algorithm decides to create a leaf node it assigns the target label from $\mathcal{Y}$ that corresponds to the most frequent one amongst the examples reaching that node. On the other hand FilterTree follows a bottom-up partition process. For both methods we experiment with hinge and logistic loss functions using different step sizes ($\{0.15, 0.25, 0.5, 0.75, 1, 2, 4, 8\}$) and up to 32 passes through the data. For the LOMTree we use different settings for final number of non-leaf nodes and for the swap resistance. Finally, for both methods we use bit precision of 30.

Table 6.4 presents the error rate for the pruning algorithm PR-M and the tree approaches LOMTree and FilterTree across all the datasets for logistic and hinge loss functions. In all cases PR-M which is based on the provided hierarchy outperforms the tree construction approaches having a large difference in their errors. The results strongly indicate that in the cases where the hierarchy is provided it is preferable to use in order to perform logarithmic prediction rather than constructing it as it leads to loss of accuracy.

7. Conclusion

We have studied in this paper flat and hierarchical classification strategies from a learning-theoretic view point in the context of large-scale taxonomies, through error generalization

5. http://hunch.net/vw

bounds of multiclass, hierarchical classifiers. The first theorem we have introduced provides an explanation to several empirical results related to the performance of such classifiers. We also introduced two methods to simplify a taxonomy by selectively pruning some of its nodes, (i) by exploiting the bound developed in the first theorem, and (ii) by designing a meta-learning technique which is based on the features derived from the approximation-error based generalization bounds proposed in Sections 5.1 and 5.2. The experimental results reported here (as well as in other papers) are in line with our theoretical developments and justify the pruning strategy adopted.

In addition to theoretically addressing the flat versus top-down classification for large-scale taxonomies, the focus of this work is also on the problem of aligning the taxonomy of classes to the set of input-output pairs. This can be useful in designing better taxonomies for large-scale classification problems. Lastly, this suggests that our theoretical development can also be exploited to grow a hierarchy of classes from a (large) set of categories, as has been done in several studies (like for example Bengio et al. (2010); Choromanska and Langford (2014)). We plan to explore this in future work.

Acknowledgments

This work was supported in part by the ANR project Class-Y, the Mastodons project Garguantua, the LabEx PERSYVAL-Lab ANR-11-LABX-0025 and the European project BioASQ (grant agreement no. 318652). The authors sincerely thank the anonymous reviewers for providing valuable feedback which has led to considerable improvement in the content of this paper.

References

Rahul Agrawal, Archit Gupta, Yashoteja Prabhu, and Manik Varma. Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In *Proceedings of the 22nd international conference on World Wide Web*, pages 13–24. International World Wide Web Conferences Steering Committee, 2013.

Rohit Babbar, Ioannis Partalas, Eric Gaussier, and Massih-Reza Amini. On flat versus hierarchical classification in large-scale taxonomies. In *Advances in Neural Information Processing Systems*, pages 1824–1832, 2013a.

Rohit Babbar, Ioannis Partalas, Eric Gaussier, and Massih-Reza Amini. Maximum-margin framework for training data synchronization in large-scale hierarchical classification. In *Neural Information Processing*, pages 336–343. Springer, 2013b.

Rohit Babbar, Cornelia Metzig, Ioannis Partalas, Eric Gaussier, and Massih-Reza Amini. On power law distributions in large-scale taxonomies. *ACM SIGKDD Explorations Newsletter*, 16(1):47–56, 2014a.

Rohit Babbar, Ioannis Partalas, Eric Gaussier, and Massih-Reza Amini. Re-ranking approach to classification in large-scale power-law distributed category systems. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 1059–1062. ACM, 2014b.

Peter L. Bartlett and Shahar Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

Peter L. Bartlett, Olivier Bousquet, and Shahar Mendelson. Local Rademacher complexities. *Annals of Statistics*, 33(4):1497–1537, 2005.

Samy Bengio, Jason Weston, and David Grangier. Label embedding trees for large multi-class tasks. In *Advances in Neural Information Processing Systems 23*, pages 163–171, 2010.

Kristin P Bennett, Nello Cristianini, John Shawe-Taylor, and Donghui Wu. Enlarging the margins in perceptron decision trees. *Machine Learning*, 41(3):295–313, 2000.

Paul N. Bennett and Nam Nguyen. Refined experts: improving classification in large taxonomies. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 11–18, 2009.

Alina Beygelzimer, John Langford, Yuri Lifshits, Gregory Sorkin, and Alexander Strehl. Conditional probability tree estimation analysis and algorithms. In *Proceedings of the Twenty-Fifth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-09)*, pages 51–58, Corvallis, Oregon, 2009a. AUAI Press.

Alina Beygelzimer, John Langford, and Pradeep D. Ravikumar. Error-correcting tournaments. In *Algorithmic Learning Theory, 20th International Conference, ALT 2009, Porto, Portugal, October 3-5, 2009. Proceedings*, pages 247–262, 2009b.

Lijuan Cai and Thomas Hofmann. Hierarchical document categorization with support vector machines. In *Proceedings* 13^{th} *ACM International Conference on Information and Knowledge Management (CIKM)*, pages 78–87. ACM, 2004.

Anna Choromanska and John Langford. Logarithmic time online multiclass prediction. *CoRR*, abs/1406.1822, 2014. URL `http://arxiv.org/abs/1406.1822`.

Bhavana Dalvi and William W Cohen. Hierarchical semi-supervised classification with incomplete class hierarchies. *In preparation*, 2014.

Ofer Dekel. Distribution-calibrated hierarchical classification. In *Advances in Neural Information Processing Systems 22*, pages 450–458, 2009.

Ofer Dekel, Joseph Keshet, and Yoram Singer. Large margin hierarchical classification. In *Proceedings of the* 21^{st} *International Conference on Machine Learning*, pages 27–35, 2004.

Jia Deng, Sanjeev Satheesh, Alexander C. Berg, and Fei-Fei Li. Fast and balanced: Efficient label tree learning for large scale object recognition. In *Advances in Neural Information Processing Systems 24*, pages 567–575, 2011.

Susan Dumais and Hao Chen. Hierarchical classification of web content. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 256–263. ACM, 2000.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

Francois Fleuret and Donald Geman. Coarse-to-fine face detection. *International Journal of computer vision*, 41(1-2):85–107, 2001.

Tianshi Gao and Daphne Koller. Discriminative learning of relaxed hierarchy for large-scale visual recognition. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2072–2079, 2011.

Siddharth Gopal and Yiming 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*, pages 257–265. ACM, 2013.

Siddharth Gopal, Yiming Yang, Bing Bai, and Alexandru Niculescu-Mizil. Bayesian models for large-scale hierarchical classification. In *Advances in Neural Information Processing Systems 25*, 2012.

Yann Guermeur. Sample complexity of classifiers taking values in $\mathbb{R}^q$, application to multiclass SVMs. *Communications in Statistics - Theory and Methods*, 39, 2010.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer New York Inc., 2001.

Daphne Koller and Mehran Sahami. Hierarchically classifying documents using very few words. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, 1997.

Balaji Krishnapuram, Lawrence Carin, Mario AT Figueiredo, and Alexander J Hartemink. Sparse multinomial logistic regression: Fast algorithms and generalization bounds. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(6):957–968, 2005.

Tie-Yan Liu, Yiming Yang, Hao Wan, Hua-Jun Zeng, Zheng Chen, and Wei-Ying Ma. Support vector machines classification with a very large-scale taxonomy. *SIGKDD Explorations*, 2005.

Hassan Malik. Improving hierarchical SVMs by hierarchy flattening and lazy classification. In *1st Pascal Workshop on Large Scale Hierarchical Classification*, 2009.

David A McAllester. Some pac-bayesian theorems. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 230–234. ACM, 1998.

Andrew McCallum, Ronald Rosenfeld, Tom M. Mitchell, and Andrew Y. Ng. Improving text classification by shrinkage in a hierarchy of classes. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 359–367, 1998.

Ron Meir and Tong Zhang. Generalization error bounds for bayesian mixture algorithms. *The Journal of Machine Learning Research*, 4:839–860, 2003.

Harikrishna Narasimhan, Harish G. Ramaswamy, Aadirupa Saha, and Shivani Agarwal. Consistent multiclass algorithms for complex performance measures. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015*, pages 2398–2407, 2015.

Andrew Y. Ng and Michael I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Advances in Neural Information Processing Systems*, pages 841–848, 2001.

Peter Norvig. *Paradigms of artificial intelligence programming: case studies in Common LISP*. Morgan Kaufmann, 1992.

Ioannis Partalas, Aris Kosmopoulos, Nicolas Baskiotis, Thierry Artières, George Paliouras, Éric Gaussier, Ion Androutsopoulos, Massih-Reza Amini, and Patrick Gallinari. LSHTC: A benchmark for large-scale text classification. *CoRR*, abs/1503.08581, 2015.

Florent Perronnin, Zeynep Akata, Zaïd Harchaoui, and Cordelia Schmid. Towards good practice in large-scale learning for image classification. In *Computer Vision and Pattern Recognition*, pages 3482–3489, 2012.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge, 2014.

Mark Schervish. *Theory of Statistics*. Springer Series in Statistics. Springer New York Inc., 1995.

Matthias Seeger. Pac-bayesian generalisation error bounds for gaussian process classification. *The Journal of Machine Learning Research*, 3:233–269, 2003.

John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, New York, NY, USA, 2004. ISBN 0521813972.

Min Sun, Wan Huang, and Silvio Savarese. Find the best path: An efficient and accurate classifier for image hierarchies. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 265–272. IEEE, 2013.

Xiaolin Wang and Bao-Liang Lu. Flatten hierarchies for large-scale hierarchical text categorization. In 5^{th} *International Conference on Digital Information Management*, pages 139–144, 2010.

Kilian Q Weinberger and Olivier Chapelle. Large margin taxonomy embedding for document categorization. In *Advances in Neural Information Processing Systems 21*, pages 1737–1744, 2008.

Gui-Rong Xue, Dikan Xing, Qiang Yang, and Yong Yu. Deep classification in large-scale text hierarchies. In *Proceedings of the* 31^{st} *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 619–626, 2008.

Yiming Yang and Xin Liu. A re-examination of text categorization methods. In *Proceedings of the* 22^{nd} *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 42–49. ACM, 1999.

J. Zhang, L. Tang, and H. Liu. Automatically adjusting content taxonomies for hierarchical classification. In *Proceedings of the* 4^{th} *Workshop on Text Mining*, 2006.