Machine Learning Strategies for Large-scale Taxonomies

Rohit Babbar

UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité : **Informatique**

Arrêté ministérial : 7 août, 2006

Présentée par

Rohit Babbar

Thèse dirigée par **Eric Gaussier**
et codirigée par **Massih-Reza Amini**

préparée au sein **Laboratoire d'Informatique de Grenoble**
et de **Ecole Doctorale Mathématiques, Sciences et Technologies de l'Information, Informatique**

Machine Learning Strategies for Large-scale Taxonomies

Thèse soutenue publiquement le **17 Octobre, 2014**,
devant le jury composé de :

Yann Guermeur
Professeur, Université de Lorraine, Rapporteur
Yiming Yang
Professeur, School of Computer Science of Carnegie Mellon University, Rapporteur
Thierry Artières
Professeur, Université Pierre et Marie Curie, Paris, Examinateur
Bernhard Schölkopf
Director, Max Planck Institute for Intelligent Systems, Tübingen, Examinateur
Denis Trystram
Professeur, Grenoble Institute of Technology, Examinateur
Eric Gaussier
Professeur, Laboratoire d'Informatique de Grenoble, Directeur de thèse
Massih-Reza Amini
Professeur, Laboratoire d'Informatique de Grenoble, Co-Directeur de thèse

ABSTRACT

In the era of Big Data, we need efficient and scalable machine learning algorithms which can perform automatic classification of Tera-Bytes of data. In this thesis, we study the machine learning challenges for classification in large-scale taxonomies. These challenges include computational complexity of training and prediction and the performance on unseen data. In the first part of the thesis, we study the underlying power-law distribution in large-scale taxonomies. This analysis then motivates the derivation of bounds on space complexity of hierarchical classifiers. Exploiting the study of this distribution further, we then design classification scheme which leads to better accuracy on large-scale power-law distributed categories. We also propose an efficient method for model-selection when training multi-class version of classifiers such as Support Vector Machine and Logistic Regression. Finally, we address another key model selection problem in large-scale classification concerning the choice between flat versus hierarchical classification from a learning theoretic aspect. The presented generalization error analysis provides an explanation to empirical findings in many recent studies in large-scale hierarchical classification. We further exploit the developed bounds to propose two methods for adapting the given taxonomy of categories to output taxonomies which yield better test accuracy when used in a top-down setup.

CONTENTS

1 INTRODUCTION 1
- 1.1 Big Data and Large-scale Learning 1
- 1.2 Challenges in Large-scale Supervised Learning 2
  - 1.2.1 Cardinality of Training and Feature set sizes 3
  - 1.2.2 Large number of Target Categories 3
  - 1.2.3 Power-law behavior of Data 4
  - 1.2.4 Exploiting Semantic Structure Among Categories 6
- 1.3 Contributions 7
- 1.4 Outline 9

2 STATE-OF-THE-ART REVIEW 11
- 2.1 Flat Classification 12
  - 2.1.1 Binary classification and One-vs-Rest 12
  - 2.1.2 Crammer-Singer Multi-class SVM 14
  - 2.1.3 Parallelizable Multinomial Logistic Regression 15
  - 2.1.4 Trace-norm for large-scale learning 16
  - 2.1.5 Other Approaches and Theoretical Studies 17
- 2.2 Hierarchical Classification 19
  - 2.2.1 Pachinko-machine based deployment of classifiers 19
  - 2.2.2 Tree-loss based optimzation 20
  - 2.2.3 Recursive Regularization 21
  - 2.2.4 Hierarchical Classification by Orthogonal Transfer 23
  - 2.2.5 Other techniques and applications of hierarchical classification 24
- 2.3 Taxonomy Adaptation 24
  - 2.3.1 Distribution Calibration 25
  - 2.3.2 Hierarchy Flattening 26
- 2.4 Taxonomy Learning 27
  - 2.4.1 Relaxed discriminative learning 27
  - 2.4.2 Fast and balanced approach to taxonomy learning 28
- 2.5 Power-law in large-scale taxonomies 29
  - 2.5.1 Training-time complexity 30
- 2.6 Conclusion 31

3 DISTRIBUTION OF DATA IN LARGE-SCALE TAXONOMIES 33
- 3.1 Introduction 33
- 3.2 Related Work 35
- 3.3 Power-law distribution in Large-scale Taxonomies 37
  - 3.3.1 Yule's model 38
  - 3.3.2 Preferential attachment models for networks and trees 41
  - 3.3.3 Model for hierarchical web taxonomies 42
  - 3.3.4 Other interpretations 44
  - 3.3.5 Limitations 45

3.3.6 Statistics per level in the hierarchy 45
3.4 Space Complexity Analysis 46
3.4.1 Relation between category size and number of features 46
3.4.2 Space Complexity of Large-Scale Classification 47
3.5 Conclusion 52

4 EXPLOITING DATA-DISTRIBUTION FOR LEARNING 55
4.1 Soft-thresholding for Classification in Power-law Distributed Categories 56
4.1.1 Power-law distribution 56
4.1.2 Related work and our contributions 58
4.1.3 Accuracy Bound on Power-law Distributed Categories 59
4.1.4 Soft-thresholding Algorithm for Higher Bound-value 60
4.1.5 Experimental Evaluation 63
4.1.6 Remarks 66
4.2 Efficient Model-selection in Big Data 67
4.2.1 Related Work 68
4.2.2 Accuracy Bound for Classification in Large Number of Categories 69
4.2.3 Using accuracy bound as alternative to k-fold cross-validation 70
4.2.4 Experimental Evaluation 72
4.2.5 Results 75
4.2.6 Remarks 76
4.3 Data-dependent Classifier Selection 76
4.3.1 Sample Complexity and LSHC 79
4.3.2 Experimental Setup 83
4.3.3 Results and Analysis 85
4.3.4 Remarks 86
4.4 Conclusion 86

5 FLAT VERSUS HIEARCHICAL CLASSIFICATION IN LARGE-SCALE TAXONOMIES 89
5.1 Introduction 89
5.2 Related Work 92
5.3 Rademacher Complexity : A Review 94
5.4 Flat vs Hierarchical Classification : A learning theoretic View-point 95
5.4.1 A hierarchical Rademacher data-dependent bound 96
5.4.2 Lowering the bound by hierarchy pruning 99
5.5 Meta-learning based pruning strategy 102
5.5.1 Asymptotic approximation error bounds for Naive Bayes 102
5.5.2 Asymptotic approximation error bounds for Multinomial Logistic Regression 105
5.5.3 A learning based node pruning strategy 108
5.6 Experimental Analysis 109
5.6.1 Flat versus Hierarchical classification 113
5.6.2 Effect of pruning 114
5.6.3 Effect of number of pruned nodes for meta-learning based pruning strategy 115
5.7 Conclusion 116

6 CONCLUSION AND PERSPECTIVES 119

LIST OF FIGURES

Figure 1 Distribution of training instances among categories for Wikipedia subset from LSHTC 5
Figure 2 Comparison of distribution of test instances in true distribution and that induced by flat SVM classifier 6
Figure 3 DMOZ and Wikipedia Taxonomies 7
Figure 4 Convex relaxations of 0-1 loss in the form of Hinge and Squared hinge loss 13
Figure 5 Spectrum of classification weight matrix **W** learned on an Imagenet subset as shown in Harchaoui et al. [2012] 17
Figure 6 Top-down deployment of SVM classifiers 20
Figure 7 Top-level flattening of hierarchy 26
Figure 8 Category size distribution for each level of the LSHTC2-DMOZ dataset. 30
Figure 9 DMOZ and Wikipedia Taxonomies 34
Figure 10 Category size vs rank distribution for the LSHTC2-DMOZ dataset. 37
Figure 11 Indegree vs rank distribution for the LSHTC2-DMOZ dataset. 38
Figure 12 Number of categories at each level in the hierarchy of the LSHTC2-DMOZ database. 38
Figure 13 Illustration of growth of taxonomy 41
Figure 14 A website is assigned to existing categories with $p(k) \propto N_k$. 43
Figure 15 (ii): Growth in categories is equivalent to growth of the tree structure in terms of in-degrees. 43
Figure 16 (iii): Growth in children categories. 43
Figure 17 Model without and with shrinking categories. In the left figure, a child category inherits all the elements of its parent and takes its place in the size distribution. 44
Figure 18 Category size distribution for each level of the LSHTC2-DMOZ dataset. 45
Figure 19 Number of features vs number of documents of each category. 46
Figure 20 Heaps' law: number of distinct words vs. number of words, and vs number of documents. 47

Figure 21 Power-law variation for features in different levels for LSHTC2-a dataset, Y-axis represents the feature set size plotted against rank of the categories on X-axis 49
Figure 22 Comparison of distribution of test instances for true distribution and that induced by flat SVM classifier 57
Figure 23 Comparison of distribution of test instances for proposed method and that induced by flat SVM classifier 64
Figure 24 Distribution of training instances among categories for Wikipedia subset from LSHTC 67
Figure 25 Variation (with λ) of cross-validation accuracy and the derived bound 74
Figure 26 Sample Taxonomy of Classes 77
Figure 27 Top-down deployment of classifiers in uniform and hybrid fashion 79
Figure 28 Variation in ratio of feature set size to training sample size 82
Figure 29 Hybird Classifier deployment using Adaptive Selection 83
Figure 30 Difference of SVM and NB accuracy 85
Figure 31 DMOZ and Wikipedia Taxonomies 90
Figure 32 The pruning procedure; the node in black is replaced by its children. 100
Figure 33 Depiction of pruning procedure 108
Figure 34 Distribution of data among classes 113
Figure 35 Accuracy performance with respect to the number of pruned nodes for `MNB` on different test sets. 116
Figure 36 Accuracy performance with respect to the number of pruned nodes for `MLR` (down) on different test sets. 117

LIST OF TABLES

Table 1 LSHTC and BioAsQ datasets and their properties 2
Table 2 Summary of notation 40
Table 3 Datasets for hierarchical classification with the properties: Number of training/test examples, target classes and size of the feature space. 51
Table 4 Model size (in GB) for flat and hierarchical models along with the corresponding values defined in Proposition 1. The symbol $\bigtriangledown$ refers to the quantity $\frac{K}{K-b(L-1)}$ 52
Table 5 LSHTC datasets and their properties 63
Table 6 Comparison of Methods 65
Table 7 Dataset description 72
Table 8 Variation in accuracy with λ parameter 75
Table 9 Training Data Properties 84
Table 10 Accuracy-Computational Complexity tradeoff 84
Table 11 Dataset description 111
Table 12 Comparison of Methods 112
Table 13 Comparison of Methods 113

1

INTRODUCTION

1.1 BIG DATA AND LARGE-SCALE LEARNING

With an increasing amount of data from various sources such as web advertizing, social media and images, automatic classification of unseen data to one of tens of thousand target classes has caught the attention of the research community. This is due to the tremendous growth in data from various sources such as social networks, web-directories and digital encyclopedias. Some of the interesting facts which emphasize the need for effective automated organization of data are the following:

- Around one thousand new articles that are added everyday to english Wikipedia
- Approximately 100 hours of video is uploaded to Youtube every minute
- Close to 20,000 of scientific articles are added to PubMed[1] every week

In order to maintain interpretability and to make these systems scalable, digital data are required to be classified among one of tens of thousands of target categories. Directory Mozilla[2], for instance, lists over 4 million websites distributed among close to 1 million categories. 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 hierarchy of the National Library of Medicine is another instance of a large-scale classification system in the domain of life sciences. In order to minimize the amount of human effort involved in such large-scale scenarios, there is a definite need to automate the process of classification of data into the target categories. To effectively address the computational barriers posed by the *Big Data*, the classical techniques of learning from data need to be adapted in order to tackle large-scale classification problems.

1 http://www.ncbi.nlm.nih.gov/pubmed
2 http://www.dmoz.org/

In the context of large-scale hierarchical classification (LSHC), open challenges like the Pascal Large Scale Hierarchical Text Classification (LSHTC) [3] and Imagenet Large Scale Visual Recognition Challenge (ILSVRC) [4] have been organized. In the domain of life-sciences, the BioAsQ challenge [5] has been organized for classifying the medical abstracts. These challenges play an important role in evaluating the current state-of-the-art techniques for large-scale classification. Table 1 shows the statistics of the various datasets released as part of the LSHTC and BioAsQ challenge.

Dataset	Training instances	Categories	Features	Parameters (in GB)
DMOZ-2010	128,710	12,294	381,580	4.3
DMOZ-2011	394,756	27,875	594,158	15.4
DMOZ-2012	383,408	11,947	348,548	3.8
SWiki-2011	456,886	36,504	346,299	11.7
LWiki-2013	2,817,603	325,056	1,617,899	489.7
BioAsQ-2013	10,876,004	26,563	444,085	10.9

Table 1: LSHTC and BioAsQ datasets and their properties

In the next section, we highlight in detail the research challenges posed by classification problems for the datasets at the scale as shown in this table.

1.2 CHALLENGES IN LARGE-SCALE SUPERVISED ARNING

Most machine learning methods and algorithms have focused primarily on datasets which are of the order of the UCI datasets A. Asuncion [2007]. However, given the scale of modern datasets as demonstrated by LSHTC datasets, the nature of classification task is quite different as compared to that for smaller datasets such as UCI. Some of the interesting research problems posed for machine learning methods involving large-scale datasets are the following:

3 http://lshtc.iit.demokritos.gr/
4 http://www.image-net.org/challenges/LSVRC/2011/
5 http://www.bioasq.org/

1.2.1 *Cardinality of Training and Feature set sizes*

The number of training examples in modern large-scale learning problems are of the order of millions. This characteristic of the data poses significant computational challenges in the following ways :

- **Scale of convex optimization problems** : The intermediate convex optimization problems involving minimizing convex surrogate losses such as Hinge loss and Logistic loss Zhang [2004b], Tewari and Bartlett [2007], Bartlett et al. [2006]are in high dimensional spaces. As a result, many off-the-shelf solvers such as LibSVM Chang and Lin [2011] run out of memory and hence cannot be applied directly. In its own right, this has led to the growth of new optimization-based techniques such as sequential dual method Keerthi et al. [2008] and trust-region based Newton method Lin et al. [2008] for large-scale learning.
- **Hyper-parameter Tuning** : Tuning the hyper-parameters such as the regularization λ parameter in Support Vector Machines Hastie et al. [2004] by the standard technique of k-fold cross-validation can be extremely computationally intensive. As another instance, on the Wikipedia-2011 dataset from the LSHTC challenge which has approximately 0.5 million training documents among 36,000 categories, 5-fold cross-validation to learn the parameter λ will take around one month on a single quad-core machine with standard hardware.

1.2.2 *Large number of Target Categories*

Learning with large number of target categories poses a relatively new challenge in machine learning as compared to large-scale learning for binary classification or classification with few tens of categories. Large-scale learning involving classification among fewer categories has been well understood theoretically Bottou and Bousquet [2008] and also stochastic version SVM solvers such as Pegasos Shalev-Shwartz et al. [2011] are available. However, learning with tens of thousand target categories involves:

- **Billions of parameters to learn** : Large-scale learning involving large number of target categories requires to learn one high dimensional weight vector for each category. For instance, for one of the LSHTC datasets, having 12,294 categories in a feature set of size 347,256 one needs to learn $12,294 \times 347,256 = 4.2$ billion parameters. In this context, the recent study by Gopal and Yang [2013a] presents a technique to learn Regularized Logistic Regression classifier by replacing the Logistic loss by an upper bound which can be easily parallelized.

- **Class imbalance** : One-vs-Rest framework, as studied in Rifkin and Klautau [2004], Allwein et al. [2001] and implemented in most modern solvers such as Liblinear Fan et al. [2008], is one of the standard methods to handle large number of categories. However, when dealing with large number of target categories makes the individual binary classification problem highly imbalanced and hence makes learning effective decision boundaries further difficult. Due to the high-dimensionality of the classification problems, conventional methods for handling class-imbalance such as those proposed in Chawla et al. [2011], Tang et al. [2009b] are not effective in large-scale problems.
- **Complexity of Inference** : For large number of target categories, the inference time becomes significantly important. For instance, to classify a test instance among K categories under the One-vs-Rest framework, one needs to evaluate $O(K)$ classifiers Harchaoui et al. [2012], Perronnin et al. [2012]. This could be significantly high for large-scale classification problems involving tens of thousand categories. Many recent works such as Bengio et al. [2010], Gao and Koller [2011], Deng et al. [2011], Yang and Tsang [2012] have focused on learning a tree-based taxonomy of categories which aim at reducing the complexity of inference to $O(\lg(K))$.
- **Universal consistency** : Another short-coming of the easily parallelizable One-vs-Rest framework is that it does not satisfy *universal consistency* property Tewari and Bartlett [2007]. On the other hand, the multi-class SVM proposed in Crammer and Singer [2002] enjoys good theoretical guarantees but is not separable into binary problems and hence not directly parallelizable.

1.2.3 *Power-law behavior of Data*

As shown in Figure 1 for the distribution of Wikipedia dataset from the LSHTC challenges, the distribution of data among categories follows power-law distribution. It has also been studied in the work of Liu et al. [2005] for large-scale web directories such as DMOZ and Yahoo! directory. Formally, let N_r denote the size of the r-th ranked category (in terms of number of documents), then :

$$N_r = N_1 r^{-\beta} \tag{1.2.1}$$

where N_1 represents the size of the 1-st ranked category and $\beta > 0$ denotes the exponent of the power law distribution. As a result, a large fraction of categories consist of very few documents in them. For instance, as discussed in Gopal and Yang [2013b], 76% of the categories in the Yahoo! directory have less than 5 documents in them and these are commonly referred to as *rare categories*. Another interpretation of this behavior is that the average number of documents

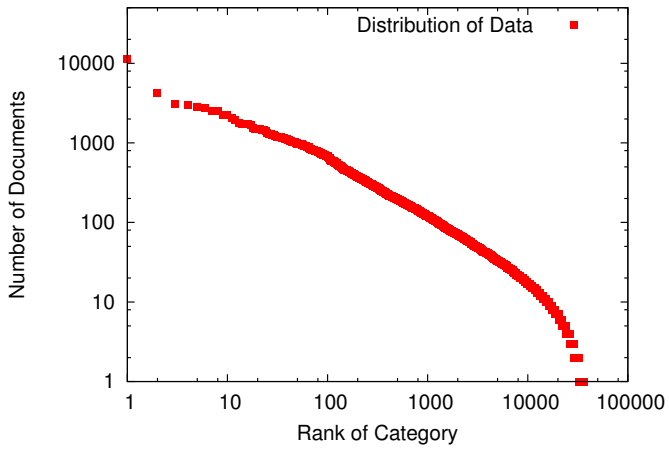


Figure 1: Distribution of 456,866 training instances (for a Wikipedia subset from LSHTC) among 36,000 categories in the training data, with X-axis representing the rank (by number of documents) of categories and Y-axis the number of documents in those categories. Approximately 15,000 of the 36,000 categories have ≤ 5 documents, with 4,000 categories having just 1 document in the training set.

per category decrease as the number of categories grow. This property of dataset leads to following problems in being to learn good classifiers:

- Due to insufficient data, it is difficult to learn good decision boundaries for rare categories.
- The class-imbalance problem is further aggravated in such power-law category systems.

As a result, a test instance which actually belongs to one of the rare categories is assigned to a bigger category. On one hand, this leads to high False Positive rate for bigger categories, and on the other hand, rare categories are lost in the classification process. This is shown for one of the datasets in Figure 2. For the distribution induced by the SVM classifier, observations in Figure 2 which demonstrate the high False-positive rate for large categories and inability to detect rare categories in such distributions are :

- On the left side of the plot, the graph for the distribution induced by the SVM classifier starts higher and remains higher as compared to true distribution, but drops much sharply on the right part, and
- Comparing the tails of the distributions on the right side of the plot, the true distribution has a fatter tail as compared to the induced distribution, i.e., it has many more categories of 1 or 2 documents as compared to the distribution induced by the SVM classifier.

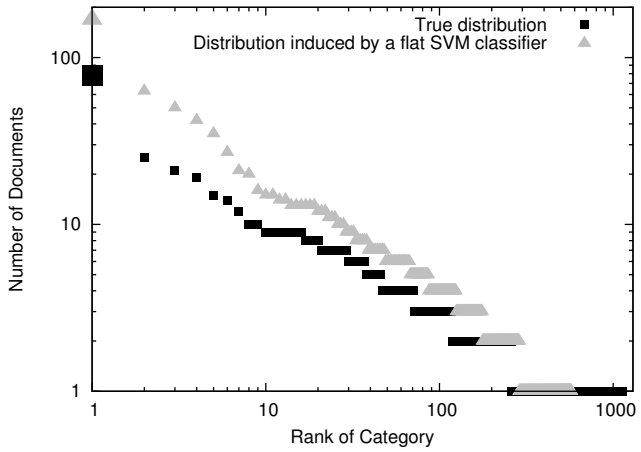


Figure 2: Comparison of distribution of test instances among categories in the true distribution and in the distribution induced by a flat SVM classifier; the X-axis represents the rank of categories (by number of documents) and Y-axis the number of documents in those categories. Categories with same number of documents effectively have same rank.

1.2.4 *Exploiting Semantic Structure Among Categories*

Typically, categories in large-scale systems have an inherent semantic structure among themselves. For instance, 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 is found in the taxonomies such as Wikipedia. The tree and DAG relationship among categories is illustrated for DMOZ and Wikipedia taxonomies in Figure 3.

Given the taxonomy structure, various approaches such as Gopal and Yang [2013b], Cai and Hofmann [2004], Dekel [2009] have been proposed which exploit this additional information differently. The taxonomy information among categories can mitigate the data-imbalance problem Babbar et al. [2013a] particularly in large-scale power-law distributed categories. Furthermore, one needs to evaluate only $O(\lg(K))$ classifiers in tree-based classifiers, also it has been shown in the work of Liu et al. [2005] that the training time complexity of hierarchical classification is lower than that for flat classification.

However, the usage of taxonomy may have some undesirable impact on the classification performance of the top-down cascade, such as:

- **Propagation Error** : Using the top-down cascade of classifiers deployed in the taxonomy leads to the propagation of classification error from top-

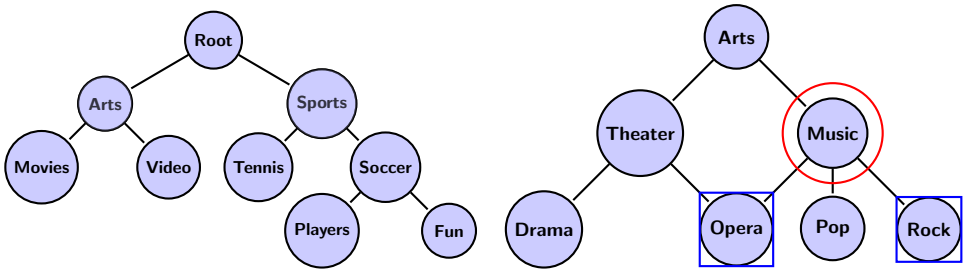


Figure 3: DMOZ and Wikipedia Taxonomies

levels towards the leaves. This cause of error is significant since the top-level categories are quite generic in nature and hence considerable overlap among them in feature space. For instance, the *Sports* and *Entertainment* nodes in Yahoo! directory are likely to a have a high degree of common vocabulary between them. The application of *Refined Experts* as studied in the work by Bennett and Nguyen [2009] aims to handle the propagation error in an effective manner.

- **Noisy Taxonomies** The taxonomy structure given a-priori as part of the training data may not be best suited to yield high classification accuracy due to the following reasons:

  1. Large-scale web taxonomies are designed with an intent of better user-experience and navigability, and not for the goal of classification.
  2. Taxonomy design is subject to certain degree of arbitrariness based on personal choices and preferences of the editors.
  3. The large-scale nature of such taxonomies poses difficulties in manually designing good taxonomies for classification.

  In the recent work by Dekel [2009] on relatively smaller taxonomies, the impact of arbitrariness on loss-function design is minimized by appropriately calibrating the edge distance between the true and predicted class. In similar spirit of taxonomy adaptation, approaches based on flattening the hierarchy such as Malik [2009], Wang and Lu [2010], have been proposed in LSHTC for large-scale settings which lead to improvement in classification accuracy as compared to using the original hierarchy.

1.3 CONTRIBUTIONS

In machine learning, a significant part of effort from a pedagogical view-point Schölkopf and Smola [2002], Bishop et al., Devroye [1996], Hastie et al. [2001] and also from the attempt to develop new methods towards addressing research challenges in machine learning Koller and Sahami [1997], McCallum et al. [1998],

Blei et al. [2003], McAllester [1998], Bousquet and Elisseeff [2002] have focused on relatively smaller sized datasets. In the light of the availability of Big data and the need to separate useful information from noise, the challenges posed by large-scale classification particularly in the presence of large number of target categories need to be addressed effectively. As discussed in the previous section, most naturally occurring large-scale datasets exhibit fit to power-law distribution and also have semantic structure among the target categories.

In this direction, we attempt to address some of the theoretical aspects of this research challenge as well as also from the view-point of developing new methods for classification in large-scale taxonomies. Specifically, our contributions in this thesis are the following:

- We first study the distribution of data in large-scale taxonomies and various generative models which give rise to the fit to power-law distribution of documents among categories in large-scale taxonomies. We refer to the famous model by Yule Yule [1925] which is governed by the assumption that a new elements joins an existing category with the probability that is proportional to its current size. In the context of large-scale taxonomies, we also study other models such as those based on Preferential attachment Barabási and Albert [1999]. We complete our analysis of power-law behavior in large-scale taxonomies by deriving an analytical form for the upper bound of space complexity of hierarchical classification technique and provide a comparison to space complexity of flat classification. This work has been published in the Special Information Group on knowledge Discovery and Data Mining (SIGKDD) Explorations Journal, 2014.
- Secondly, we exploit the distribution of data in large-scale category systems to address the three challenges for classification, (i) classification accuracy, (ii) training time via model selection and hyper-parameter tuning, and (iii) prediction time. Addressing the problem depicted in Figure 2 which is faced by most state-of-the-art methods, we propose a simple but non-trivial upper bound on the accuracy of a classifier which classifies instances among tens of thousand power-law distributed categories. Our soft-thresholding based method for ranking target categories by their posterior probabilities is published in Special Information Group on Information Retrieval (SIGIR) 2014 Babbar et al. [2014]. Exploiting the accuracy upper bound further, we also demonstrate efficient method for model-selection as an alternative to computationally expensive k-fold cross-validation. Using the sample complexity bounds for discriminative and generative classifiers as derived in Ng and Jordan [2001], we also propose a method to combine Support Vector Machine and Naive Bayes classifiers in a top-down cascade which leads to faster training and prediction in large-scale hierarchical classification. This work Babbar et al. [2012]

and its variant Partalas et al. [2012] were published in Conference on Information and Knowledge Management (CIKM) 2012, and International Conference on Neural Information Processing (ICONIP) 2012 respectively.

- Lastly, we address the problem of flat versus hierarchical classification in large-scale taxonomies from a learning theoretic point of view. The goal in this problem is to learn from the training data to choose one of strategies, (i) use flat classification, i.e., ignore the given taxonomy structure altogether, or (ii) perform hierarchical classification with classifiers deployed in a top-down cascade. This research challenge, even though fundamental to the nature of classification problem in large-scale taxonomies, has not been addressed earlier from a learning-theoretic aspect. To our knowledge, our work Babbar et al. [2013a] in Neural Information Systems (NIPS) 2013, was the first such attempt towards this problem wherein we developed Rademacher complexity based generalization error bounds to study this problem. In order to handle the noisy taxonomies, we further exploit the developed bounds for designing techniques using which the given can be adapted to learn a new taxonomy which leads to better classification accuracy. This can also be viewed as synchronization of two parts of the training data, (i) in the form of input, output pairs $\langle \mathbf{x}, y \rangle$, and (ii) as given by the taxonomy. This work was published in International Conference on Neural Information Processing (ICONIP) 2013. The work presented in this chapter is currently under revision after first round of reviews from Journal of Machine Learning Research (JMLR).

1.4 OUTLINE

The brief outline of the this thesis is as follows:

- In Chapter 2, we review the current state-of-the-art for large-scale supervised classification for flat and hierarchical classification. Even though, our focus is primarily on mono-label classification throughout the thesis, we also briefly mention some of the multi-label approaches for large-scale classification.
- We present in Chapter 3, various generative models which lead to the fit to power-law distribution of documents among categories in large-scale taxonomies. We also present an analytical study of the space complexity of hierarchical classification.
- In Chapter 4, we also derive non-trivial upper bound on the accuracy of a classifier which is particularly useful in large-scale power-law distributed

categories. Based on this upper-bound, we propose techniques for better classification accuracy and efficient model selection.

- In Chapter 5, we present the learning theoretic bounds for top-down hierarchical classification and address the flat versus hierarchical classification problem in large-scale taxonomies. We also propose two methods for taxonomy adaptation by hierarchy pruning which is shown to yield better classification accuracy than the hierarchy of classes given a-priori.
- Finally, we conclude this thesis and present some of the perspectives.

2

STATE-OF-THE-ART REVIEW

Classification of data into large-number of categories has assumed considerable significance over the last few years. This is due to considerable growth in data from various sources such as social media, commercial products and descriptions, images data from uploaded photos and videos, and from collaborative encyclopedias. For instance, enterprises such as Amazon and ebay have product hierarchies which are aimed at providing easy access to customers for searching the desired product and also other products which are closely related to itself. Furthermore, motivated by the challenge of fine-grained classification in the context of images, classification into large number of categories has become quite important.

As a result, the process of automatic classification is no longer restricted to small scale datasets with two or few tens of labels. In view of emerging commercial interests in large-scale problems and also public availability of such datasets, recent research interest in machine learning for tens of thousand target categories has increased considerably. This is also evident from large number of scientific publications in large-scale learning and big data every year which address various aspects of large-scale learning. Furthermore, big data has been the theme of many conferences and workshops in the recent years.

It is important to note that by large-scale learning we refer to large-number of target categories and focus on classification challenges arising out of such machine learning setting. By large-scale learning, we do not imply problem settings with binary classification problem such as when spam versus non-spam classification for a large corpus is performed. Even though classification for binary problem or with few tens of target categories on large datasets are interesting and have been studied (from the point of view of stochastic training) theoretically (Bottou and Bousquet [2008], Zhang [2004a]) and empirically (Shalev-Shwartz et al. [2011]).

Going beyond the classical problem in machine learning of designing a classifier with low generalization error, other metrics of evaluation such as prediction time, training time, and space complexity of the model become important in order the assess the quality of a classifier. The immediate approach to handle large number of categories is to consider them as many independent binary classification problems as the number of target categories, which is also referred

to as *One-versus-Rest* as discussed primarily in Rifkin and Klautau [2004], Allwein et al. [2001]. For SVM classifier, the method proposed by Weston [1998] to handle multi-class problems is by adding constraints for every category and thereby the number of constraints grow quadratically with number of target categories. Another approach for handling multi-class problems which is based on the generalized notion of margin for multi-class problems is proposed in Crammer and Singer [2002].

However, these multi-class approaches have prediction time which is linear in the of number of categories, i.e., $O(K)$ for K categories. For large number of target categories, in the range of tens of thousand, it is desired to have prediction time which is sublinear in the number of categories. Typically, for large number of categories, there exists a semantic structure among categories in the form of rooted tree or a directed cyclic graph. This can be viewed in the form of parent-child relationship which also depicts a transition from general categories to special categories when one traverses the path from root towards the leaves. In the light of the inherent existence of the semantic structure among categories, there has been significant research focus on hierarchical classification systems. In the next sections, we discuss the state-of-the-art methods for large-scale learning. Since flat classification is a special case of hierarchical classification in which case the taxonomy structure is ignored, we give below the more general formulation in terms of setup for hierarchical classification.

2.1 FLAT CLASSIFICATION

Flat approaches to large-scale learning ignore the hierarchical structure among the categories. This makes them simpler to interpret and implement. However, these approaches may suffer from data-imbalance problem particularly in the presence of power-law distributed category systems.

2.1.1 *Binary classification and One-vs-Rest*

Most recent studies have focused on Support Vector Machines (SVM) and Logistic Regression (LR) for large-scale learning. These discriminative learning algorithms minimize a combination of empirical error and model complexity. The template of the objective function which is minimized is of the following form:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} R_{emp}(\mathbf{w}) + \lambda \; Reg(\mathbf{w}) \tag{2.1.1}$$

where $Reg(\mathbf{w})$ is the regularization term to avoid complex models and $R_{emp}(.)$ represents the empirical error.

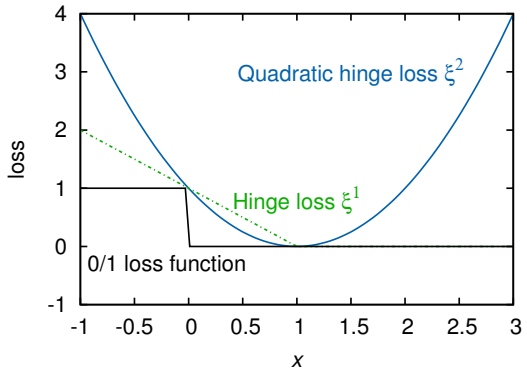


Figure 4: Convex relaxations of 0-1 loss in the form of Hinge and Squared hinge loss

In binary classification, the training set is of the form $(\mathbf{x}_i, y_i), i = 1 \ldots m, y_i \in \{-1, +1\}$. For SVM classifier, the 0-1 loss $R_{emp}(.)$ is replaced by its convex surrogate called the hinge-loss which is given by $(\max(0, 1 - y_i \mathbf{w}^T \mathbf{x}_i))$. For Logistic Regression $R_{emp}(.)$, the convex surrogate is based on logistic loss $(\log(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)))$. $Reg(w)$ is typically of the form $\frac{1}{2}\mathbf{w}^T\mathbf{w}$, unless sparse solution is desired in which case it is replaced by $|\mathbf{w}|$. The hyper-parameter λ controls the trade-off between the empirical error and regularization term.

More specifically, the optimization problem for learning binary L2-regularized, L1-loss SVM classifier is given by

$$\min_{\mathbf{w}} \frac{\lambda}{2}||\mathbf{w}||^2 + \sum_{i=1}^{m}(\max(0, 1 - y_i \mathbf{w}^T \mathbf{x}_i))$$

On similar lines, the L2-regularized, L2-loss SVM classifier

$$\min_{\mathbf{w}} \frac{\lambda}{2}||\mathbf{w}||^2 + \sum_{i=1}^{m}(\max(0, 1 - y_i \mathbf{w}^T \mathbf{x}_i))^2$$

The L1-loss and L2-loss relaxations are shown in Figure 2.1.1. The L2-regularized, Logistic Regression classifier is given by

$$\min_{\mathbf{w}} \frac{\lambda}{2}||\mathbf{w}||^2 + \sum_{i=1}^{m}\log(1 + exp(-y_i \mathbf{w}^T \mathbf{x}_i))$$

To handle multi-class problems under the One-vs-Rest framework, one binary classifier which is parameterized by the weight vector $\mathbf{w}_k$ is learnt for each of the K target categories. The training data is transformed K times for the construction of each binary problem such that while learning $\mathbf{w}_k$ the training instances which belong to category k are labeled $+1$ and all the other training

instances are labeled -1. At inference time, to estimate the target category of instance $\mathbf{x}$, the predicted category is the one which satisfies $\arg\max_k \mathbf{w}_k^T\mathbf{x}$. This approach has the following salient features:

- It is simple to interpret and more importantly, easily parallelizable which is a desirable property for training classifiers in settings with large number of target categories.
- It has been shown in the work of Rifkin and Klautau [2004], that when the binary classifiers are properly calibrated, the One-vs-Rest classifier can perform at par with other approaches such as One-versus-One and approaches based on Error Correcting Output Codes (Dietterich and Bakiri [1995]).
- A major drawback One-vs-Rest framework is that it does not satisfy *universal consistency* property Tewari and Bartlett [2007] and hence does not enjoy strong theoretical guarantees.

2.1.2 *Crammer-Singer Multi-class SVM*

The approach studied in Crammer and Singer [2002] proposed a more natural way to handle to multi-class problem instead of considering them as independent binary problems. For given training data in the form of instance-label pairs $(\mathbf{x}_i, y_i), i = 1 \ldots m, y_i \in \{1 \ldots K\}$, the formulation of the optimization problem under this framework is given by

$$\min_{\mathbf{w}_k, \xi_i} ||\mathbf{w}_k||^2 + C \sum_{i=1}^{m} \xi_i$$

The constraints for the above optimization problem are given by, $\forall i = 1 \ldots m$

$$\mathbf{w}_{y_i}^T\mathbf{x}_i - \mathbf{w}_k^T\mathbf{x}_i \geq 1 - e_i^k - \xi_i, \text{ and } \xi_i \geq 0$$

where

$$e_i^k = \begin{cases} 1 & \text{if } y_i = k \\ 0 & \text{otherwise} \end{cases}$$

The decision function is given by

$$\arg\max_{k=1\ldots K} \mathbf{w}_k^T\mathbf{x}$$

The primal optimization problem as given above is typically solved from its dual formulation. The dual is given by the following

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \tfrac{1}{2}\sum_{k=1}^{K}||\mathbf{w}_k||^2 + \sum_{i=1}^{m}\sum_{k=1}^{K} e_i^k \alpha_i^k \\ \text{subject to} \quad & \sum_{k=1}^{K}\alpha_i^k = 0, \ \ \forall i = 1\dots m \\ & \alpha_i^k \leq C_{y_i}^k \forall i = 1\dots m, \forall k = 1\dots K \end{aligned} \tag{2.1.2}$$

where

$$\mathbf{w}_k = \sum_{i=1}^{m} \alpha_i^k \mathbf{x}_i \forall k, \ \boldsymbol{\alpha} = [\alpha_1^1 \dots \alpha_1^K, \dots, \alpha_m^1 \dots \alpha_m^K]^T$$

and

$$C_{y_i}^k = \begin{cases} 0 & \text{if } y_i \neq k \\ C & \text{otherwise} \end{cases}$$

Sequential dual method for solving the dual optimization problem in (mc-svm-dual-chap2) was proposed in Keerthi et al. [2008] for handling large-scale problems.

Unlike the one-vs-rest framework of handling multi-class problems, this formulation has strong theoretical guarantees such as universal consistency Tewari and Bartlett [2007], Zhang [2004b], Bartlett et al. [2006]. However, it suffers from two major disadvantages in the context of large-scale learning :

- Since it learns the parameters $\mathbf{w}_k$ simultaneously for each target category, it is not inherently parallelizable, and hence may lead to extremely high training time. In a typical large-scale setting, since the dimensionality of each $\mathbf{w}_k$ is of the order of hundreds of thousand, and for a classification problem involving few tens of thousand categories, the total number of parameters are in the range of billions. Therefore, being able to parallelize the training procedure is highly desirable property.
- Furthermore, the memory requirements of this method are quite high as the tasks cannot be split across categories.

2.1.3 *Parallelizable Multinomial Logistic Regression*

To handle the drawbacks mentioned for the multi-class SVM in the formulation proposed by Crammer-Singer, the recent study in Gopal and Yang [2013a] proposes a method to parallelize the optimization of the objective function. For regularized multinomial logistic regression, the probability for instance $\mathbf{x}$ to belong to category k is given by

$$P(y = k|\mathbf{x}) = \frac{exp(\mathbf{w}_k^T \mathbf{x})}{\sum_{k'=1}^{K} exp(\mathbf{w}_{k'}^T \mathbf{x})}$$

Let $\mathbf{W} = \{\mathbf{w}_1, \ldots, \mathbf{w}_K\}$ denote the matrix of weight vectors, then the training objective in this framework is given by

$$\min_{\mathbf{W}} \frac{\lambda}{2} \sum_{k=1}^{K} ||\mathbf{w}_k||^2 - \sum_{k=1}^{K} \sum_{i=1}^{N} e_i^k \mathbf{w}_k^T \mathbf{x}_i + \sum_{i=1}^{N} \log(\sum_{k'=1}^{K} exp(\mathbf{w}_{k'}^T \mathbf{x})) \tag{2.1.3}$$

As such the objective function given by the above equation is not parallelizable due to the coupling of all the class-level parameters together inside a log-sum-exp function. The authors use the concavity of the log-function to replace the objective in 2.1.3 by a parallelizable version. The log concavity bound is given by

$$\log(\gamma) \leq a\gamma - \log(a) - 1, \ \forall \gamma, a > 0$$

Using the above bound and introducing parameters a_i for each training instance, the log-partition function for instance i is bounded as follows :

$$\log(\sum_{k=1}^{K} exp(\mathbf{w}_k^T \mathbf{x}_i)) \leq a_i \sum_{k=1}^{K} exp(\mathbf{w}_k^T \mathbf{x}_i) - \log(a_i) - 1$$

From the above substitution and denoting by $\mathbf{a}$ the vector of $a_i, i = 1 \ldots m$, the new objective function is given by

$$\min_{\mathbf{W},\mathbf{a}} \frac{\lambda}{2} \sum_{k=1}^{K} ||\mathbf{w}_k||^2 + \sum_{i=1}^{N} \left[- \sum_{k=1}^{K} e_i^k \mathbf{w}_k^T \mathbf{x}_i + a_i \sum_{k=1}^{K} exp(\mathbf{w}_k^T \mathbf{x}_i) - \log(a_i) - 1 \right] \tag{2.1.4}$$

The above objective is parallelizable, even though non-convex. However, the authors shows that the new obejctive function in Equation 2.1.4 has many desirable properties such that it can be exploited to obtain the solution to the original ojective function in Equation 2.1.3.

2.1.4 *Trace-norm for large-scale learning*

Another important insight for large-scale classification in the context of image data has been done in Harchaoui et al. [2012], wherein the authors perform singular value decomposition on the matrix of weight vectors $\mathbf{W}$ and show that it has a rank which is much lower than K. Motivated by this observation, they propose a learning objective which captures the low-rank embedding of the target categories. This is achieved by adding a low-rank enforcing penalty in the form of trace norm regularization to the Frobenius norm penalty. The learning objective considered in this work is of the form

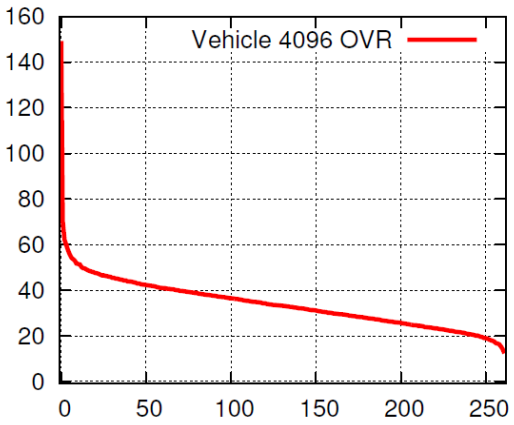


Figure 5: Spectrum of classification weight matrix $\mathbf{W}$ learned on an Imagenet subset as shown in Harchaoui et al. [2012]

$$\min_{\mathbf{W}} \quad \lambda_1 \text{rank}(\mathbf{W}) + \lambda_2 ||\mathbf{W}||^2 + R_m(\mathbf{W}) \tag{2.1.5}$$

where $R_m(\mathbf{W})$ denotes the empirical risk

$$R_m(\mathbf{W}) = \frac{1}{m} \sum_{i=1}^{m} L(\mathbf{W}; \mathbf{x}_i, y_i)$$

and the authors take multi-class logistic loss as the loss function to compute the empirical loss which has the following form

$$L(\mathbf{W}; \mathbf{x}, y) = \log(1 + \sum_{k \in 1 \ldots K/y} exp\{\mathbf{w}_k^T \mathbf{x} - \mathbf{w}_y^T \mathbf{x}\})$$

Since the objective function in 2.1.5 is non-smooth and non-convex, it is relaxed by replacing the rank$(\mathbf{W})$ by its tightest convex surrogate i.e., the trace norm. The new objective function is given by

$$\min_{\mathbf{W}} \quad \lambda_1 ||\mathbf{W}||_{\sigma,1} + \lambda_2 ||\mathbf{W}||^2 + R_m(\mathbf{W}) \tag{2.1.6}$$

where $||\mathbf{W}||_{\sigma,1}$ denotes the trace-norm of $\mathbf{W}$. The objective function in the above equation is convex but is non-differentiable due to the low-rank enforcing penalty. The authors demonstrate the similarity of this objecitve to sparse logistic regression for binary problems Hastie et al. [2001].

2.1.5 *Other Approaches and Theoretical Studies*

The work in Perronnin et al. [2012] is based on using One-versus-Rest classifier for large-scale image data from ILSVRC challange. They authors propose

some important recommendations for using One-vs-Rest strategy for image classification to achieve state-of-the-art performance using dense Fisher Vector representation of images Sánchez et al. [2013], Perronnin and Dance [2007]. The proposed recommendations include:

- Learning with stochastic gradient descent is well suited for large-scale datasets
- Early stopping can be used as an effective mechanism to achieve regularization
- A small-enough step-size w.r.t. the learning rate is sufficient for good performance

In the recent study by Weston et al. [2013], the authors propose a label partitioning technique for sub-linear ranking involving large-number of labels. Another quite recent study for large-scale classification have been studied in the work of Gupta et al. [2014], wherein the authors propose to approximate the expected error with a different empirical loss called the *empirical class-confusion loss*. For the large-scale online training, they show that an online empirical class-confusion loss can be implemented for stochastic gradient descent by ignoring stochastic gradients corresponding to a repeated confusion between classes.

From a learning theoretic view-point, the work in Daniely et al. [2012] compares various multi-class approaches including multi-class SVM, One-vs-Rest, One-vs-One and tree-based classifiers Beygelzimer et al.. Some of the important findings of this study are the following:

- The estimation errors of One-vs-Rest, multi-class SVM, and tree-based classifiers are approximately close to each other,
- The authors prove that the hypothesis class of multi-class SVM essentially contains the hypothesis classes of both One-vs-Rest and tree-based classifiers. Furthermore, these inclusions are strict and since the estimation errors of these three methods are roughly the same, it follows that the multi-class SVM method dominates both One-vs-Rest and tree-based classifiers in terms of achievable prediction performance, and
- They also show that the hypothesis class of One-vs-One essentially contains the hypothesis class of multi-class SVM, and that there can be a substantial gap in the containment.

The work in Guermeur [2007] also provides important theoretical insight into the VC-theory for multi-class classification. In the context of large-scale multi-label classification various approaches such as Agrawal et al. [2013], Yu et al. [2013], Hariharan et al. [2010], Cisse et al. [2013].

2.2 HIERARCHICAL CLASSIFICATION

In hierarchical classification, in addition to the input-output pairs, we are also given the taxonomy of classes which represents the underlying semantic structure. Formally, we use the following setup for understanding hierarchical classification and the approaches proposed to handle such problems.

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 sisters $\mathfrak{S}(v) = \{v' \in V \setminus \{\perp\}; v \neq v' \wedge \pi(v) = \pi(v')\}$ and its daughters $\mathfrak{D}(v) = \{v' \in V \setminus \{\perp\}; \pi(v') = v\}$. 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

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

Given a new test instance $\mathbf{x}$, the goal is to predict the class $\widehat{y}$. In top-down hierarchical classification, the classifier (such as SVM) is learnt at every decision node in the tree as is shown in Figure 2.2.1. The various state-of-the-art methods differ in the way they learn the classifier at each node. 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.

2.2.1 *Pachinko-machine based deployment of classifiers*

In Pachinko-machine based top-down deployment of classifiers the decisions are made at each level of the hierarchy. This method selects the best class at each level of the hierarchy and iteratively proceeds down the hierarchy until a leaf node is reached. This is typically done by making a sequence of predictions iteratively in a top-down fashion starting from the root . At each non-leaf node $v \in V$, a score $f_c(\mathbf{x}) \in \mathbb{R}$ is computed for each daughter $c \in \mathfrak{D}(v)$ and the child $\widehat{c}$ with the maximum score is predicted i.e. $\widehat{c} = \arg\max_{c:(v,c)\in E} f_c(\mathbf{x})$.

For SVM classifier, $f_c(\mathbf{x})$ is modeled as a linear classifier such that $f_c(\mathbf{x}) = \mathbf{w}_c^T \mathbf{x}$. To learn a one-versus-rest L2-regularized, L2-loss SVM-based discriminative

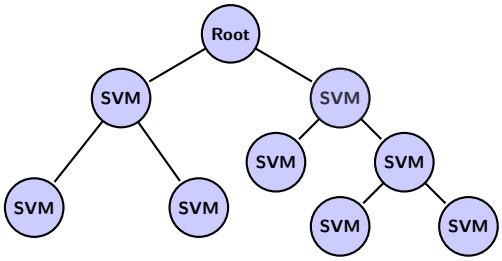


Figure 6: Top-down deployment of SVM classifiers

classifier for node v, the following optimization problem is solved for each daughter c of v

$$\min_{\mathbf{w}_c, \xi} \frac{\lambda}{2}||\mathbf{w}_c||^2 + \sum_{i=1}^{m_v} \xi_{(i,c)}^2$$

The indices i above are such that $\forall i, 1 \leq i \leq m_v, y_i \in L_v$, were L_v denotes the set of leaves in the subtree rooted at node v and m_v denotes the number of training examples for which the root-to-leaf path passes through the node v. Furthermore, if $y_i \in L_c$ and $(v, c) \in E$, then the constraints for the above optimization problem are given by, $\forall i$

$$\mathbf{w}_c^t \mathbf{x}_i \geq 1 - \xi_{(i,c)}, \quad \text{and } \xi_{(i,c)} \geq 0$$

This method has the advantage that it is faster to train and is very natually parallelizable owing to the independence of optimization problems at each node in the taxonomy. Furthermore, due to the tree-nature of the problem, the number of predictors that one needs to evaluate is logarithmic in the number of target categories. This method is shown to yield competetive performance on large-scale datasets as shown in the work of Liu et al. [2005], Dumais and Chen [2000]. However, many variants of this methods have been proposed recently.

2.2.2 *Tree-loss based optimzation*

In the recent work of Bengio et al. [2010], the authors observe that in a hierarchical setup, the final prediction can be wrong due to mis-classification at *any* node in the root to leaf path. This is unlike the Panchinko machine model in which each mis-classification at every node is accounted individually. With this insight as the motivation, they propose a tree-loss based optimization wherein the slack variable is shared acorss all nodes along each of the root to leaf path in the tree.

Denoting by $b_j(\mathbf{x})$ as the index of the best node (w.r.t. to the decision function) in the hierarchy at depth j Specifically, the loss function, called *tree loss*, on the training data is given by

$$R_{emp}(f_{tree}) = \frac{1}{m}\sum_{i=1}^{m}\max_{j\in B(\mathbf{x})} I(y_i \in l_j)$$

where $B(\mathbf{x}) = \{b_1(\mathbf{x}) \dots b_{D(\mathbf{x})}(\mathbf{x})\}$ and $D(\mathbf{x})$ denotes the depth in the tree for final prediction of instance $\mathbf{x}$. Assuming that internal nodes of the tree are indexed by j and l_j denotes the set of leaves under the sub-tree rooted at node j.

Replacing the 0-1 loss function in the form of indicator function and adding the 2-norm regularizer, the optimization objective is given by

$$\sum_{j=1}^{|V|}\left(\gamma||\mathbf{w}_j||^2 + \frac{1}{m}\sum_{i=1}^{m}\xi_{ij}\right) \tag{2.2.1}$$

such that

$$\forall i,j \begin{cases} C_j(y_i)f_j(\mathbf{x}_i) \geq 1 - \xi_{ij} \\ \xi_{ij} \geq 0 \end{cases}$$

where $C_j(y_i) = 1$ if $y_i \in l_j$ and -1 otherwise.

To take into account the tree loss, the above optimization as given Equation 2.2.1 is modified by introducing a slack variable which is shared across all the decision nodes for a given training instance. This leads to the following tree-loss based optimization objective

$$\gamma\sum_{j=1}^{|V|}||\mathbf{w}_j||^2 + \frac{1}{m}\sum_{i=1}^{m}\xi_i \tag{2.2.2}$$

such that

$$f_r(\mathbf{x}_i) \geq f_s(\mathbf{x}_i) + 1 - \xi_i, \forall r,s : y_i \in l_r \wedge y_i \notin l_s \wedge (\exists p : (p,r) \in E \wedge (p,s) \wedge E) \tag{2.2.3}$$

$$\xi_i \geq 0, \forall i = 1 \dots m \tag{2.2.4}$$

2.2.3 *Recursive Regularization*

In this recently proposed strategy Gopal and Yang [2013b], the hierarchy structure is incorporated into the optimization problem in the form a regularizer. In the hierarchical approaches, the weight vector is required to be learnt at each node of the hierarchy tree. Therefore, let the matrix $\mathbf{W}$ be such that its columns represent the weight vectors at each of the decision nodes, i.e., $\mathbf{W} = \{\mathbf{w}_v, v \in V\}$. The regularization term for the optimization problem at each node is such that it encourages the weight vector of a node to be close

to that of its parent node. The framework is proposed for SVM and Logistic Regression classifier, and for SVM is given by the following:

HR-SVM

$$\min_{W} \sum_{v \in V} \frac{1}{2} ||\mathbf{w}_v - \mathbf{w}_{\pi}(v)||^2 + C \sum_{v \in \mathcal{Y}} \sum_{i=1}^{m} (1 - C_v(y_i) \mathbf{w}_v^T \mathbf{x}_i)_+ \tag{2.2.5}$$

For each non-leaf node $v \notin \mathcal{Y}$, differentiating (2.2.5) wrt $\mathbf{w}_v$, it leads to a closed-form update for $\mathbf{w}_v$, which is given by

$$\mathbf{w}_v = \frac{1}{|\mathfrak{D}(v)| + 1} \left(\mathbf{w}_{\pi}(v) + \sum_{c \in \mathfrak{D}(v)} \mathbf{w}_c \right) \tag{2.2.6}$$

where $\mathfrak{D}(v)$ denotes the daughters of node v.

For each leaf node $y \in \mathcal{Y}$, the following is solved:

$$\min_{\mathbf{w}_y} \frac{1}{2} ||\mathbf{w}_y - \mathbf{w}_{\pi(y)}||^2 + C \sum_{i=1}^{m} \xi_{iy} \tag{2.2.7}$$

subject to

$$\xi_{iy} \geq 0, \xi_{iy} \geq 1 - C_y(y_i) \mathbf{w}_y^T \mathbf{x}_i, \ \forall i = 1 \dots m$$

The above optimization is solved by dual co-ordinate descent as proposed in Hsieh et al. [2008]. The dual of the above optimization problem is given by

$$\min_{\boldsymbol{\alpha}} \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j C_y(y_i) C_y(y_j) \mathbf{x}_i^T \mathbf{x}_j - \sum_{i=1}^{m} \alpha_i (1 - C_y(y_i) \mathbf{w}_{\pi(y)}^T \mathbf{x}_i), \ \forall i = 1 \dots m \tag{2.2.8}$$

$$0 \leq \alpha_i \leq C, \ \forall i = 1 \dots m \tag{2.2.9}$$

The update for each dual variable in the above optimization problem can be derived in closed form. This can be derived by substituting α_i in equation (2.2.3) by $\alpha_i + d$ and dropping all terms which do not depend on d, and then solving the following problem in one variable.

$$\min_{d} \frac{1}{2} d^2 (\mathbf{x}_i^T \mathbf{x}_i) + d \left(\sum_{i=1}^{m} \alpha_i C_y(y_i) \mathbf{x}_i \right)^T \mathbf{x}_i - d(1 - C_y(y_i) \mathbf{w}_{\pi(y)}^T \mathbf{x}_i) \tag{2.2.10}$$

$$0 \leq \alpha_i + d \leq C \tag{2.2.11}$$

For Logistic Regression classifier at the inner nodes of the tree, the optimization problem is given by the following :

HR-LR

$$\min_{W} \sum_{v \in V} \frac{1}{2} ||\mathbf{w}_v - \mathbf{w}_{\pi}(v)||^2 + C \sum_{v \in \mathcal{Y}} \sum_{i=1}^{m} \log(1 + exp(-C_v(y_i)\mathbf{w}_v^T \mathbf{x}_i) \quad (2.2.12)$$

The update for each non-leaf node is same as in HR-SVM. For the leaf nodes y, the objective function is given by

$$\min_{\mathbf{w}_y} \frac{1}{2} ||\mathbf{w}_y - \mathbf{w}_{\pi(y)}||^2 + C \sum_{i=1}^{m} \log(1 + \exp(-C_y(y_i)\mathbf{w}_y^T \mathbf{x}_i)) \quad (2.2.13)$$

The gradient of the above can computable in the closed form and is given by

$$G = \mathbf{w}_y - \mathbf{w}_{\pi(y)} - C \sum_{i=1}^{m} \frac{1}{1 + \exp(C_y(y_i)\mathbf{w}_y^T \mathbf{x}_i)} C_y(y_i)\mathbf{x}_i \quad (2.2.14)$$

Since the gradient can be computed in closed-form it is possible to directly apply quasi newton methods such as Limited-memory Broyden-Fletcher-Goldfarb-Shanno (LBFGS) Liu and Nocedal [1989] to solve the above optimization problem. The authors also propose a fast and easily parallelizable method which can exploit a parallel computing infrastructure such as Hadoop.

2.2.4 *Hierarchical Classification by Orthogonal Transfer*

Unlike the work presented in Gopal and Yang [2013b], Cai and Hofmann [2004], which is based on the similarity of parameters for parent-child pair of nodes, another line of work which is based on notion of dis-similarity between parent-child pairs is studied in Xiao et al. [2011]. In this strategy, the authors propose to add a regularization terms which tends to encourage the weight vector of a child node to be different from that its ancestor. Given the training set $\{(\mathbf{x}_i, y_i)\}_{i=1}^{m}$ and the taxonomy $\mathcal{H} = (V, E)$ of categories such that nodes (except the root) of the taxonomy are indexed from 1 to $|v - 1|$, the optimization problem to learn the weight vector at the internal nodes is given by

$$\sum_{v=1}^{|V|-1} \mathbf{K}_{vv} ||\mathbf{w}_v|| + \sum_{v=1}^{|V|-1} \sum_{v' \in \mathfrak{P}(v)} \mathbf{K}_{vv'} |\mathbf{w}_v^T \mathbf{w}_{v'}| + \frac{C}{m} \sum_{i=1}^{m} \xi_i \quad (2.2.15)$$

The constraints are given by the following

$$\mathbf{w}_v^T \mathbf{x}_i - \mathbf{w}_{v'}^T \mathbf{x}_i \geq 1 - \xi_i, \ (\forall v' \in \mathfrak{S}(v), \forall v \in \mathfrak{P}(y_i), \forall i = 1 \ldots m) \quad (2.2.16)$$

$$\xi_i \geq 0, \forall i = 1 \ldots m \quad (2.2.17)$$

The terms $\mathbf{K}_{vv'}|\mathbf{w}_v^T\mathbf{w}_{v'}|$ encourage the weight vector of parent-child node pairs to be orthogonal to each other by penalizing the dot product between the weight vectors. The entering of symmetric matrix K are chosen as follows:

$$\mathbf{K}_{vv'} = \begin{cases} |\mathfrak{D}(v)| + 1 & \text{if } v = v' \\ \alpha & \text{if } v \in \mathfrak{P}(v') \\ 0 & \text{otherwise} \end{cases}$$

where α is set to 1 to make the problem convex Boyd and Vandenberghe [2009]. The authors propose a regularized dual averaging method Nesterov [2009] for solving the above optimization problem.

2.2.5 *Other techniques and applications of hierarchical classification*

The authors in Cissé et al. [2012] use the hierarchical information to learn compact binary codes for the categories by using auto-encoder based architecture for learning the representation. The induced binary problems are empirically shown to be easier than those induced by the randomly generated codes by ECOC giving competitive performances compared to classical One-versus-Rest method and ECOC. An incremental reranking based framework for hierarchical classification has been proposed for small-scale problem involving Reuters Corpus Volume 1 (RCV1) in the work by Ju and Moschitti [2013]. The reranker technique exploits category dependencies, which allow it to recover from the propagation errors while its top-down structure results in faster training and prediction time.

Hierarchical classification has also been studied for multi-label problem in the works such as Bi and Kwok [2012b, 2011, 2012a]. Many recent studies have applied hierarchical classification in variety of domains in order to tackle large-scale problems. Hierarchical classification in the context of e-commerce has been studied by using cost-sensitive penalties in the work by Chen and Warren [2013]. The recent work in Ren et al. [2014] proposes to employ a multi-label hierarchical classification frame-work for classification of social text streams, wherein the authors address the challenges of concept drift, short-text and complicated relationships among category labels. A recent study wherein the authors study various evaulation measures for hierarchical classification is given in Kosmopoulos et al. [2013].

2.3 TAXONOMY ADAPTATION

Various approaches for hierarchical classification such as Cai and Hofmann [2004], Dekel et al. [2004] utilize the distance in the hierarchy to design the loss

function such that loss incurred on a mis-classification is proportional to the distance in the hierarchy between the actual and predicted label. Hence, the classifiers are designed to minimze the regularized version of this loss function. However, as studied in Dekel [2009], the distance in the tree may not be a good representation of the difference between the true and predicted label. The given hierarchy may have non-uniformity and unbalanced nature due to the following reasons:

1. Large-scale web taxonomies are designed with an intent of better user-experience and navigability, and not for the goal of classification.
2. Taxonomy design is subject to certain degree of arbitrariness based on personal choices and preferences of the editors.
3. The large-scale nature of such taxonomies poses difficulties in manually designing good taxonomies for classification.

2.3.1 *Distribution Calibration*

As a result, Dekel [2009] proposed a distribution calibrated approach in which the underlying distribution over labels is used to set the edge weights in a way that adds balance to the taxonomy and compensates for arbitrariness in taxonomy design. For each $y \in \mathcal{Y}$, let $p(y)$ denote the marginal probability of the label y in the distribution $\mathcal{D}$. For each $v \in V$, define $p(v) = \sum_{y \in \mathcal{Y} \cap \tau(v)} p(y)$, where $\tau(v)$ denotes the set of all nodes which are in the subtree rooted at node v. Unlike the work in Dekel et al. [2004], where each edge is weighted with unit weight for computing the tree-distance loss, the edge between nodes v and $\pi(v)$ in the distribution calibrated framework is given by $\log(p(\pi(v))/p(v))$. The weighted tree-distance loss between the labels y and y' is given by the following:

$$l(y, y') = 2\log(p(\lambda(y, y'))) - \log(p(y)) - \log(p(y')) \tag{2.3.1}$$

where $\lambda(y, y')$ represents the lowest common ancestor in the tree of the leaf nodes y and y'. Based on this loss-function, the authors propose a calibrated definition of statistical risk for hierarchical classification. For a classifier f, its risk is given by $R(f) = \mathbb{E}_{\mathbf{x} \times y \sim \mathcal{D}}\left[l(f(\mathbf{x}), y)\right]$. Defining $q(f, v) = \mathbb{P}(f(\mathbf{x}) = v)$ which denotes the probability that f outputs node v, when $\mathbf{x}$ is drawn according to the marginal distribution of $\mathcal{D}$ over $\mathcal{X}$. and $r(f, v) = \mathbb{P}(\lambda(f(\mathbf{x}), y) = v)$, the probability that the lowest common ancestor of $f(\mathbf{x})$ and y is v when $(\mathbf{x} \times y) \sim \mathcal{D}$. The risk $R(f)$ can be re-written using Equation 2.3.1 as the following:

$$R(f) = \sum_{v \in V} (2r(f, v) - q(f, v)) \log(p(v)) - \sum_{y \in \mathcal{Y}} p(y) \log(p(y)) \tag{2.3.2}$$

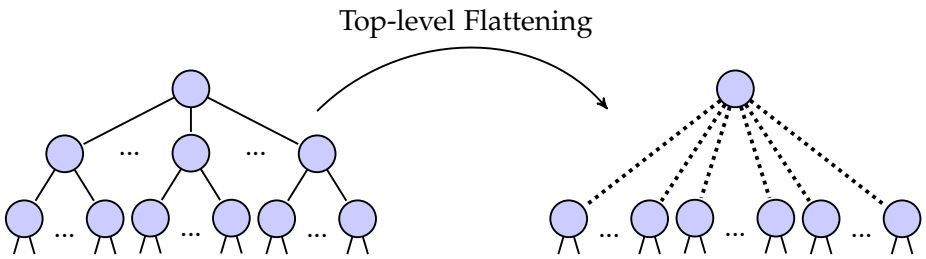


Figure 7: Top-level flattening of hierarchy

The second term in the above equation (denoted by $H(\mathcal{Y})$) represents the Shannon entropy of the label distribution and is independent of the classifier f. Assuming that the sample size is infinite and harmonic number h_n is defined by $h_n = \sum_{i=1}^{n} \frac{1}{i}$, with $h_0 = 0$. Defining the following variables :

$$A_i = \min\{j \in \mathbb{N} : y_{i+j} \in \tau(f(\mathbf{x}_i))\} - 1$$

$$B_i = \min\{j \in \mathbb{N} : y_{i+j} \in \tau(\lambda(f(\mathbf{x}_i), y_i))\} - 1$$

A_1+2 is the index of the first example after $(\mathbf{x}_1, y_1)$ whose label is contained in the subtree rooted at $f(\mathbf{x}_1)$, and B_1+2 is the index of first example whose label is contained in the subtree rooted at $\lambda(f(\mathbf{x})_i, y_i)$. Writing $\bar{R}(f) = R(f) - H(\mathcal{Y})$ and $L_1 = h_{A_1} - 2h_{B_1}$, the authors show that L_1 is an unbiased estimator for $\bar{R}(f)$. Furthermore, they also present technique for reducing the variance of this estimator and present an algorithmic reduction from hierarchical classification to cost-sensitive classification.

2.3.2 *Hierarchy Flattening*

In view of the arguments given in the previous section about the susceptibility of the given hierarchy to noise and arbitrariness, there is a need to exploit the information provided by the large-scale hierarchy in a more cautious manner. The given taxonomy $\mathcal{H} = (V, E)$ can be altered in some ways to maintain the original hierarchical relationship such as by removing a node v and directly connecting $\pi(v)$ to $\mathfrak{D}(v)$. A particular case of altering the taxonomy has been studied in the works of Malik [2009], Wang and Lu [2010] wherein the authors propose to remove certain layers in the taxonomy and replacing the nodes in that layer by their children. This is also shown in figure 7 wherein the first layer is flattened. Authors in Malik [2009], Wang and Lu [2010] show that flattening can lead to improvement in classification at the cost more training time. However, they provide no formal framework on which layer to flatten and how to identify which need to be flattened. This is one of the key problems which will study in this thesis and present theoretically well-founded approaches to identify the nodes to prune.

2.4 TAXONOMY LEARNING

While dealing with large-number of target categories, many recent works have focused on the problem of learning the taxonomy when no taxonomy is given a-priori. This is particularly important from the point of view of computational complexity of prediction. One of the initial studies in this direction is conducted in Beygelzimer et al. [2009], wherein the authors present an online algorithm for learning the hierarchical structure with local probability estimators at internal nodes of the induced hierarchy. We next discuss two approaches for learning the hierarchical structure which have been quite successful in addressing this research challenge.

2.4.1 *Relaxed discriminative learning*

In this strategy Gao and Koller [2011], the hierarchical structure and the local classifiers at the induced nodes are learnt jointly in two-step iterative procedure which is similar in spirit to the Expectation-Maximization paradigm. For an induced node v, suppose l_v denotes the set of leaves under sub-tree rooted at v. The first step involves splitting l_v into two easily distinguishable mutually exclusive subsets S_y^+ and S_y^-. Relaxed learning implies that those categories which are not easily separable are put in the set S_y^0. The three mutually exclusive category subsets are colored with coloring variables $u_k \in \{-1, 0, +1\}$ In the second step, assuming the induced split, the parameters of the binary classifier at each node are learnt using margin-based algorithm such as SVM. Given the input training set $\{(\mathbf{x}_i, y_i)\}_{i=1}^m$, the local training data S_v at each node consists of $S_x^+ = \{\mathbf{x}_i : y_i \in S_y^+\}$ and $S_x^- = \{\mathbf{x}_i : y_i \in S_y^-\}$.

In order to encourage balanced splits and taxonomies with unusally high depths which will increase the model complexity considerably, the authors propose the optimization problem at each node :

$$\begin{aligned} \min_{\mathbf{w},b,\{\mu_k\},\{\xi_i\}} \quad & \tfrac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^m |\mu_{y_i}|\xi_i - A\sum_{i=1}^m |\mu_{y_i}| \\ \text{subject to} \quad & \mu_k \in \{-1, 0, +1\} \forall k \in \mathcal{Y} \\ & \mu_{y_i}(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \forall i \\ & \xi_i \geq 0, \forall i \\ & -B \leq \sum_{k=1}^{|} \mathcal{Y}|\mu_k \leq B \\ & \sum_{k=1}^{|\mathcal{Y}|} \mathbb{1}\{\mu_k > 0\} \geq 1 \text{ and } \sum_{k=1}^{|\mathcal{Y}|} \mathbb{1}\{\mu_k < 0\} \geq 1 \end{aligned} \tag{2.4.1}$$

The last term in the objective function $-A\sum_{i=1}^{m}|\mu_{y_i}|$ encourages more categories to be part of the binary classification problem, in order to avoid trivial solutions. The third constraint is aimed at achieving balanced splits, while the last contraint enforces that each split consists of atleast one positive and one negative category. The above optimization problem is solved in an EM-like fashion, but fixing the coloring in the first step, and learning the binary SVM. In the second step, after having learnt the weight vector $\mathbf{w}$, the coloring problem is solved. By using the framework as given in Cristianini [1998], Platt et al. [1999], Bennett et al. [2000], the authors also provide theoretical guarantees on the generalization performance of their algorithm.

2.4.2 *Fast and balanced approach to taxonomy learning*

This approach proposed in Deng et al. [2011], attempts to learn jointly the split among categories and the paramters of the classifier at that node. This is formulated as a problem of maximizing the accuracy of a local classifier subject to efficieny constraints. The efficiency is measured in terms of ambiguity which is defined as the size of the label set l_v of the node v with respect to the size its parent's label set $l_{\mathfrak{P}(v)}$.

Let at the current node v, let Q be the pre-specified branching factor and $K = |l_v|$. Also let P denote the splits of v which can be also be seen as a partition matrix, i.e. $P \in \{0,1\}^{Q\times K}$ such that $P_{qk} = 1$ if category k appears in the label set of the child q, and $P_{qk} = 0$ otherwise. For each child $q \in \mathfrak{D}(v)$, there exists a one-vs-rest binary classifier, which therefore leads to a matrix with Q columns denoted $\mathbf{W}$.

At node v, such that the given training instance $\mathbf{x}, y$ such that $y \in l_v$, let $\hat{v} = \arg\max_{q\in\mathfrak{D}(v)} f_q(x)$ be the winning child. For parameters $\mathbf{W}$ and P, the loss at the current node is $L(\mathbf{W}, \mathbf{x}, y, P) = 1 - P_{\hat{q}y}$. When the partitions are fixed, this leads to a one-vs-rest multiclass problem at v. Practically, a regularized version of of the following convex relaxation is solved :

$$\tilde{L}(\mathbf{W}, \mathbf{x}_i, y_i, P) = \max_{q\in A_i, r\in B_i} \{\mathbf{w}_r^T\mathbf{x}_i - \mathbf{w}_q^T\mathbf{x}_i\} \tag{2.4.2}$$

The efficiency of the hierarchy measured in terms of the ambiguity constraints which encourage balanced partitions. For a given example $(\mathbf{x}, y)$ and paramters P and W, the ambiguity is given by

$$A(\mathbf{W}, \mathbf{x}, P) = \frac{1}{K}\sum_{k=1}^{K} P(\hat{q}, k) \tag{2.4.3}$$

The final optimization problem consisting of accuracy and efficiency constraints is given by

$$\begin{array}{rc} \text{minimize}_{\mathbf{w},P} & \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{m}\tilde{L}(\mathbf{W},\mathbf{x}_i,y_i,P) \\ \text{subject to} & \frac{1}{m}\sum_{i=1}^{m}A(\mathbf{W},\mathbf{x},P) \leq \epsilon \\ & P \in \{0,1\}^{Q\times K} \end{array} \tag{2.4.4}$$

The proposed algorithm iteratively minimizes the classification error and ambiguity at each node. The integer constraints are replaced by continuous range relaxation and rounding, which utlimately leads to good performance on datasets drawn from the ILSVRC challenge.

2.5 POWER-LAW IN LARGE-SCALE TAXONOMIES

In order to study the growth process of large-scale taxonomies, model based on preferential attachment are most appropriate. This model is based on the seminal model by U. Yule Yule [1925], originally formulated for the taxonomy of biological species. It applies to systems where elements of the system are grouped into classes, and the system grows both in the number of classes, and in the total number of elements (which are here documents or websites). In its original form, Yule's model serves as explanation for power law formation in any taxonomy, irrespective of an eventual hierarchy among categories. Similar dynamics have been applied to explain scaling in the connectivity of a network, which grows in terms of nodes and edges via preferential attachment Barabási and Albert [1999]. Recent further generalizations apply the same growth process to trees Klemm et al. [2005], Geipel et al. [2009], Tessone et al. [2011].

Power-law behavior in large-scale web taxonomies was first studied in the work by Yang et al. [2003], Liu et al. [2005] wherein the authors empirically show that the distribution of documents among categories at each level of the hierarchy exhibits fit to the power-law distribution. As a result, a large fraction of categories consist of very few documents in them.

Let N_{lr} denote the size of the r-th ranked category (in terms of number of documents), then :

$$N_{lr} = N_{l1}r^{-\beta_l} \tag{2.5.1}$$

where N_{l1} represents the size of the 1-st ranked category at level l and $\beta_l > 0$ denotes the exponent of the power law distribution at this level. The fit of the distribution of documents among categories to the power-law distribution (for

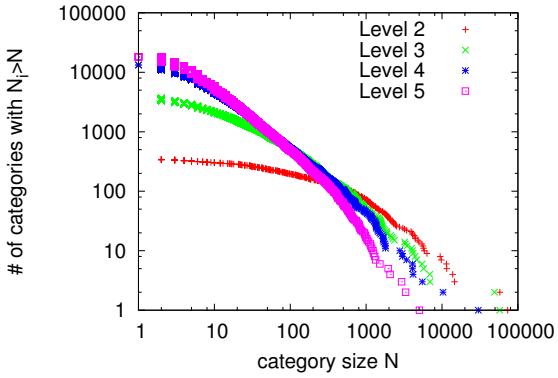


Figure 8: Category size distribution for each level of the LSHTC2-DMOZ dataset.

various levels) is also shown in Figure 8 for the DMOZ dataset derived from LSHTC challenge.

2.5.1 *Training-time complexity*

The authors in Liu et al. [2005] compare the training time complexity of flat and hierarchical classification techniques. As shown in Joachims [1999], Platt [1999], the computational complexity of training SVM grows super-linearly with number of training instances. Therefore, for flat classification under the one-vs-rest setup, with m documents in the training set which are distributed among K categories, the training time complexity is given by

$$Q^{flat} = K \times O(m^c), \quad c > 1 \tag{2.5.2}$$

The training time complexity of hierarchical SVM is given by

$$Q^{hier} = \sum_{l=1}^{L} \sum_{r=1}^{m_l} b_l \times O(m_{lr}^c) \tag{2.5.3}$$

where b_l is the branching factor at level l and m_{lr} is the number of documents in the r-th ranked category at level l. Using Equation(2.5), Liu et al. [2005]

show that the computational complexity of training hierarchical SVM is upper bounded by

$$
\begin{aligned}
Q^{hier} &\leq \left(b_0 + \sum_{l=1}^{L} \frac{\frac{b_l}{c\beta_l - 1}\left(c\beta_l - m_l^{1-c\beta_l}\right)}{\left[\frac{1}{\beta_l - 1}\left(1 - (m_l+1)^{1-\beta_l}\right)\right]^c} \right) \times O(m^c) \\
&= \frac{1}{m}\left(b_0 + \sum_{l=1}^{L} \frac{\frac{b_l}{c\beta_l - 1}\left(c\beta_l - m_l^{1-c\beta_l}\right)}{\left[\frac{1}{\beta_l - 1}\left(1 - (m_l+1)^{1-\beta_l}\right)\right]^c} \right) \times Q^{flat}
\end{aligned}
\tag{2.5.4}
$$

where L is the total number of levels in the taxonomy, β_l is the power-law exponent at level l and b_l denotes the average branching factor at level l.

The authors also empirically demonstrated that on the Yahoo! taxonomy $Q^{flat} \approx 600 \times Q^{hier}$. Essentially, they concluded that for large-scale datasets involving tens of thousand categories, training flat SVM is virtually infeasible without a distributed computing infrastructure.

2.6 CONCLUSION

Large-scale learning with tens of thousand target categories is an interesting research direction and gaining increasing attention in academic and industrial research. In this chapter, we presented state-of-the-art approaches proposed to handle large-number of target categories. These include approaches which exploit the semantic structure (hierarchical approaches) and those which ignore this information (flat techniques). We also mentioned in detail some of the successful techniques for building taxonomies when no semantic structure is provided a-priori. We also discussed the relationship of power-law distribution in large-scale web-taxonomies. One of the major contribution of this thesis is to study in more detail role of this distribution and exploit it for designing effective classification strategies.

3

DISTRIBUTION OF DATA IN LARGE-SCALE TAXONOMIES

In many of the large-scale physical and social complex systems phenomena fat-tailed distributions occur, for which different generating mechanisms have been proposed. In this chapter, we study models of generating power law distributions in the evolution of large-scale taxonomies such as Open Directory Project, which consist of websites assigned to one of tens of thousands of categories. The categories in such taxonomies are arranged in tree or DAG structured configurations having parent-child relations among them. We first quantitatively analyze the formation process of such taxonomies, which leads to power law distribution as the stationary distributions. In the context of designing classifiers for large-scale taxonomies, which automatically assign unseen documents to leaf-level categories, we then highlight how the fat-tailed nature of these distributions can be leveraged to analytically study the space complexity of hierarchical top-down classifiers. We then compare the space complexity of flat versus hierarchical classifiers, both empirically and analytically. In this respect, this study complements earlier works which have compared the computational complexity of training time for these two classification strategies.

3.1 INTRODUCTION

With the tremendous growth of data on the web from various sources such as social networks, online business services and news networks, structuring the data into conceptual taxonomies leads to better scalability, interpretability and visualization. Yahoo! directory, the open directory project (ODP) and Wikipedia are prominent examples of such web-scale taxonomies. The Medical Subject Heading hierarchy of the National Library of Medicine is another instance of a large-scale taxonomy in the domain of life sciences. The taxonomies consist of classes arranged in a hierarchical structure with parent-child relations among them and can be in the form of a rooted tree or a directed acyclic graph.

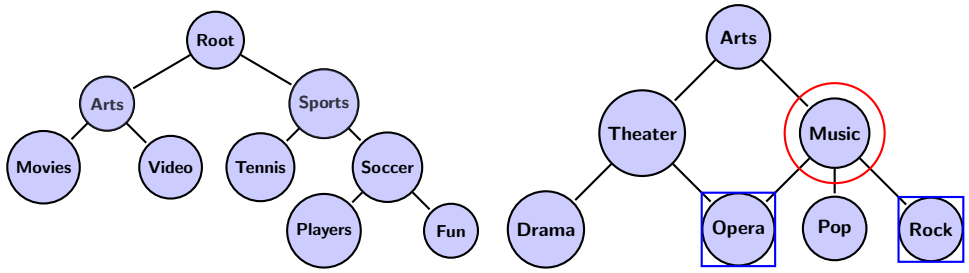


Figure 9: DMOZ and Wikipedia Taxonomies

ODP for instance, is in the form of a rooted tree, lists over 5 million websites distributed among close to 1 million categories and is maintained by close to 100,000 human editors. Wikipedia, on the other hand, represents a more complicated directed graph taxonomy structure consisting of over a million categories. In this context, large-scale hierarchical classification deals with the task of automatically assigning labels to unseen documents from a set of target classes which are represented by the leaf level nodes in the hierarchy. The tree and DAG relationship among categories is illustrated for DMOZ and Wikipedia taxonomies in Figure 3.1.

In this chapter, we study the distribution of data and the hierarchy tree in large-scale taxonomies with the goal of modeling the process of their evolution. This is undertaken by a quantitative study of the evolution of large-scale taxonomy using models of preferential attachment, based on the famous model proposed by Yule Yule [1925] and showing that throughout the growth process, the taxonomy exhibits a fat-tailed distribution. We apply this reasoning to both category sizes and tree connectivity in a simple joint model. Formally, a random variable X is defined to follow a power law distribution if for some positive constant a, the (complementary cumulative distribution) is given as follows:

$$P(X > x) \propto x^{-a}$$

Power law distributions are found in a wide variety of physical and complex social systems, ranging from city population, distribution of wealth to citations of scientific articles Newman [2005a]. It is also found in network connectivity, where the internet and wikipedia are prominent examples Song et al. [2005], Capocci et al. [2006]. Our analysis in the context of large-scale web-taxonomies not only leads to better visualization of such large-scale data, but we further leverage this additional *meta-information* to present a concrete analysis of space complexity for hierarchical classification. In order to tackle the challenges posed by ever increasing scale of training data size in terms of number of documents, feature set size and number of target classes, space complexity of the trained classifiers plays a crucial role in the applicability of classification systems in many applications of practical importance.

The space complexity analysis provides an analytical comparison of the trained model for hierarchical and flat classification, which can be used to select the appropriate model a-priori for the classification problem at hand, without actually having to train any models. Exploiting the power law nature of taxonomies to study the training time complexity for hierarchical Support Vector Machines has been performed in Yang et al. [2003], Liu et al. [2005]. The authors therein justify the power law assumption only *empirically*, unlike our analysis in Section 3.3 wherein we describe the generative process of large-scale web taxonomies more concretely, in the context of similar processes studied in other models. Despite the important insights of Yang et al. [2003], Liu et al. [2005], space complexity has not been treated formally so far.

The rest of this chapter is organized as follows. Related work on reporting power law distributions and on large scale hierarchical classification is presented in Section 3.2. In Section 3.3, we recall important growth models and quantitatively justify the formation of power laws as they are found in hierarchical large-scale web taxonomies by studying the evolution dynamics that generate them. Building on the explanation for the class size distribution in terms of distribution of websites, we then appeal to Heaps' law in Section 3.4.1, to explain the distribution of features among categories which is then exploited in Section 3.4 for analyzing the space complexity for hierarchical classification schemes. The analysis is empirically validated on publicly available DMOZ datasets from the Large Scale Hierarchical Text Classification Challenge(LSHTC)[1] and patent data (IPC) [2] from World Intellectual Property Organization.

3.2 RELATED WORK

Power law distributions are reported in a wide variety of physical and social complex systems Newman [2005b]. Furthermore, it has been shown in the work of Faloutsos et al., Capocci et al. [2006] that internet topologies exhibit power laws with respect to the in-degree of the nodes. Also the size distribution of large-scale web category systems, measured in terms of number of websites, exhibits a fat-tailed distribution, as empirically demonstrated in Yang et al. [2003], Liu et al. [2005] for the Open Directory Project (ODP). Various models have been proposed for generating power law distributions, a phenomenon that may be considered fundamental in complex systems as the normal distribution in statistics Richmond and Solomon [2001]. However, in contrast to the derivation of normal distribution via the central limit theorem, models explaining power law formation all rely on an approximation. Some explanations are based on

1 http://lshtc.iit.demokritos.gr/
2 http://web2.wipo.int/ipcpub/

phase transitions or on multiplicative noise Wilson and Kogut [1974], Takayasu et al. [1997].

In order to study the growth process of large-scale taxonomies, model based on preferential attachment are most appropriate. This model is based on the seminal model by U. Yule Yule [1925], originally formulated for the taxonomy of biological species. It applies to systems where elements of the system are grouped into classes, and the system grows both in the number of classes, and in the total number of elements (which are here documents or websites). In its original form, Yule's model serves as explanation for power law formation in any taxonomy, irrespective of an eventual hierarchy among categories. Similar dynamics have been applied to explain scaling in the connectivity of a network, which grows in terms of nodes and edges via preferential attachment Barabási and Albert [1999]. Recent further generalizations apply the same growth process to trees Klemm et al. [2005], Geipel et al. [2009], Tessone et al. [2011]. In this body of work, we explain an approximate power-law in the child-to-parent category relations by the model proposed by Klemm et al. [2005]. Furthermore, we combine this formation process in a simple manner with the original Yule model in order to explain also a power law in category sizes, i.e. we provide a comprehensive explanation for the formation process of large-scale web taxonomies such as DMOZ.

In addition to prediction accuracy, other metrics of performance such as prediction and training speed as well as space complexity of the model have become increasingly important. This is especially true in the context of challenges posed by problems in the space of Big Data, wherein an optimal trade-off among such metrics is desired. The significance of prediction speed in such scenarios has been highlighted in recent studies such as Bengio et al. [2010], Gao and Koller [2011], Partalas et al. [2012], Bottou and Bousquet [2008]. The prediction speed is directly related to space complexity of the trained model, as it may not be possible to load a large trained model in the main memory due to sheer size. In order to study the space complexity of large-scale hierarchical classifiers, we further infer a third scaling distribution for the number of features per category. This is done via the empirical Heaps's law Egghe [2007], which consists of a scaling law between text length and the size of its vocabulary. Despite its direct impact on prediction speed, no earlier work has focused on space complexity of hierarchical classifiers.

Some of the earlier works on exploiting hierarchy among target classes for the purpose of text classification have been studied in Koller and Sahami [1997], Cai and Hofmann [2004] and Dekel et al. [2004] wherein the number of target classes were limited to a few hundreds. However, the work by Liu et al. [2005] is among the pioneering studies in hierarchical classification towards addressing web-scale directories such as Yahoo! directory consisting of over 100,000 target

classes. The authors analyze the performance with respect to accuracy and training time complexity for flat and hierarchical classification. Additionally, while the existence of power law distributions has been used for analysis purposes in Yang et al. [2003], Liu et al. [2005] no thorough justification is given on the existence of such phenomenon. Our analysis in Section 3.3, attempts to address this issue in a quantitative manner. More recently, other techniques for large-scale hierarchical text classification such as Bennett and Nguyen [2009], Xue et al. [2008], Gopal et al. [2012] have been proposed.

3.3 POWER-LAW DISTRIBUTION IN LARGE-SCALE TAXONOMIES

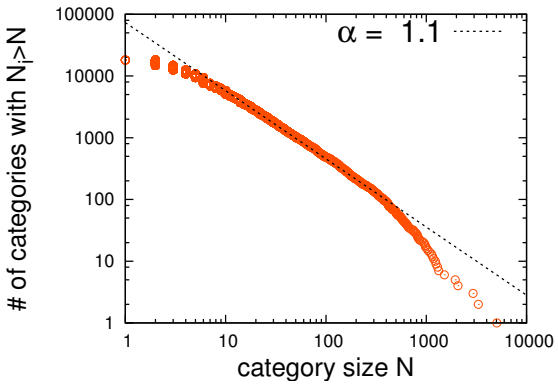


Figure 10: Category size vs rank distribution for the LSHTC2-DMOZ dataset.

We begin by introducing the complementary cumulative size distribution for category sizes. Let N_k denote the size of category k (in terms of number of documents), then the probability that $N_k > N$ is given by

$$P(N_k > N) \propto N^{-\beta} \tag{3.3.1}$$

where $\beta > 0$ denotes the exponent of the power law distribution[3]. Empirically, it can be assessed by plotting the rank of a category's size against its size (see Figure 10). The derivative of this distribution, the category size probability density $p(N_k)$, also follows a power law with exponent $(\beta + 1)$, i.e. $p(N_k) \propto N_k^{-(\beta+1)}$.

Two of our empirical findings are a power law for both the complementary cumulative category size distribution and the counter-cumulative in-degree

3 To avoid confusion, we denote the power law exponents for in-degree distribution and feature size distribution γ and δ.

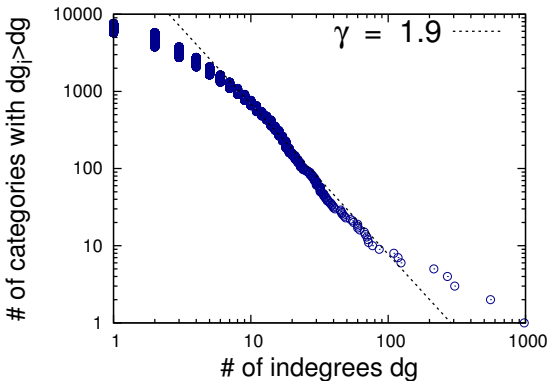


Figure 11: Indegree vs rank distribution for the LSHTC2-DMOZ dataset.

distribution, shown in Figures 10 and 11, for LSHTC2-DMOZ dataset which is a subset of ODP. This dataset[4] contains 394,000 websites and 27,785 categories. The number of categories at each level of the hierarchy is shown in Figure 12.

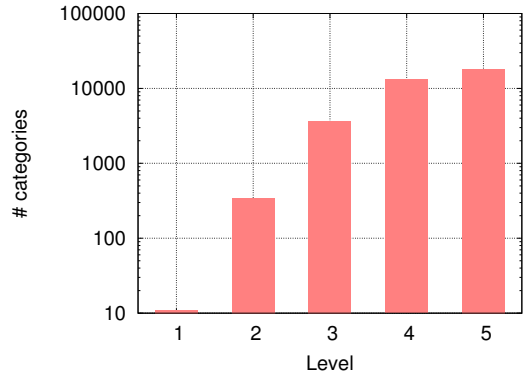


Figure 12: Number of categories at each level in the hierarchy of the LSHTC2-DMOZ database.

We explain the formation of these two laws via models by Yule Yule [1925] and a related model by Klemm Klemm et al. [2005], detailed in sections 3.3.1 and 3.3.2, which are then related in section 3.3.3.

3.3.1 *Yule's model*

Yule's model describes a system that grows in two quantities, in elements (documents or websites in case of web directories such as DMOZ) and in classes to which the elements are assigned. It assumes that for a system having κ

4 http://lshtc.iit.demokritos.gr/LSHTC2_datasets

classes, the probability that a new element will be assigned to a certain class, say k, is proportional to its current size,

$$p(k) = \frac{N_k}{\sum_{k'=1}^{\kappa} N_{k'}} \tag{3.3.2}$$

It further assumes that for every m elements that are added to the pre-existing classes in the system, a new class of size 1 is created[5].

The described system is constantly growing in terms of elements and classes, so strictly speaking, a stationary state does not exist Mandelbrot [1959]. However, a stationary distribution, the so-called Yule distribution, has been derived using the approach of the master equation with similar approximations by Simon [1955], Newman [2005a], Klemm et al. [2005]. Here, we follow Newman Newman [2005a], who considers as one time-step the duration between creation of two consecutive classes. From this follows that the average number of elements per class is always $m+1$, and the system contains $\kappa(m+1)$ elements at a moment where the number of classes is κ. Let $p_{N,\kappa}$ denote the fraction of classes having N elements when the total number of classes is κ. Between two successive time instances, the probability for a given pre-existing class i of size N_i to gain a new element is $mN_i/(\kappa(m+1))$. Since there are $\kappa\, p_{N,\kappa}$ classes of size N, the expected number such classes which gain a new element (and grow to size $(N+1)$) is given by :

$$\frac{mN}{\kappa(m+1)}\,\kappa\, p_{N,\kappa} = \frac{m}{(m+1)}\,N\, p_{N,\kappa} \tag{3.3.3}$$

The number of classes with N websites are thus fewer by the above quantity, but some which had $(N-1)$ websites prior to the addition of a new class have now one more website. This step depicting the change of the state of the system from κ classes to $(\kappa+1)$ classes is shown in Figure 13. Therefore, the expected number of classes with N documents when the number of classes is $(\kappa+1)$ is given by the following equation:

$$(\kappa+1)p_{N,(\kappa+1)} = \kappa\, p_{N,\kappa} + \frac{m}{m+1}[(N-1)p_{(N-1),\kappa} - Np_{N,\kappa}] \tag{3.3.4}$$

The first term in the right hand side of equation 3.3.4 corresponds to classes with N documents when the number of classes is κ. The second term corresponds to the contribution from classes of size $(N-1)$ which have grown to size N, this is shown by the left arrow (pointing rightwards) in Figure 13. The last term corresponds to the decrease resulting from classes which have gained an element and have become of size $(N+1)$, this is shown by the right arrow

5 The initial size may be generalized to other small sizes; for instance Tessone et al. [2011] consider entrant classes with size drawn from a truncated power law.

Variables	
N_k	Number of elements in class k
κ	Number of classes
$p_{N,\kappa}$	Fraction of classes having N elements when the total number of classes is κ
Constants	
m	Number of elements added to the system after which a new class is added
Indices	
k	Index for the class

Table 2: Summary of notation

(pointing rightwards) in Figure 13. The equation for the class of size 1 is given by:

$$(\kappa+1)p_{1,(\kappa+1)} = \kappa\, p_{1,\kappa} + 1 - \frac{m}{m+1}p_{1,\kappa} \tag{3.3.5}$$

As the number κ of classes (and therefore the number of elements $\kappa(m+1)$) in the system increases, the probability that a new element is classified into a class of size N, given by Equation (3.3.3), is assumed to remain constant and independent of κ. Under this hypothesis, the stationary distribution for class sizes can be determined by solving equation (3.3.4) and using equation (3.3.5) as the initial condition. This is given by

$$p_N = (1+1/m)B(N, 2+1/m)$$

where $B(.,.)$ is the beta distribution. It has been termed *Yule distribution* Simon [1955]. Written for a continuous variable N, it has a power law tail:

$$p(N) \propto N^{-2-\frac{1}{m}}$$

From the above equation the exponent of the density function is between 2 and 3. Its cumulative size distribution $P(N_k > N)$, as given by equation (4.1.1), has an exponent given by

$$\beta = (1+(1/m)) \tag{3.3.6}$$

which is between 1 and 2. The higher the frequency $1/m$ at which new classes are introduced, the bigger β becomes, and the lower the average class size. This exponent is stable over time although the taxonomy is constantly growing.

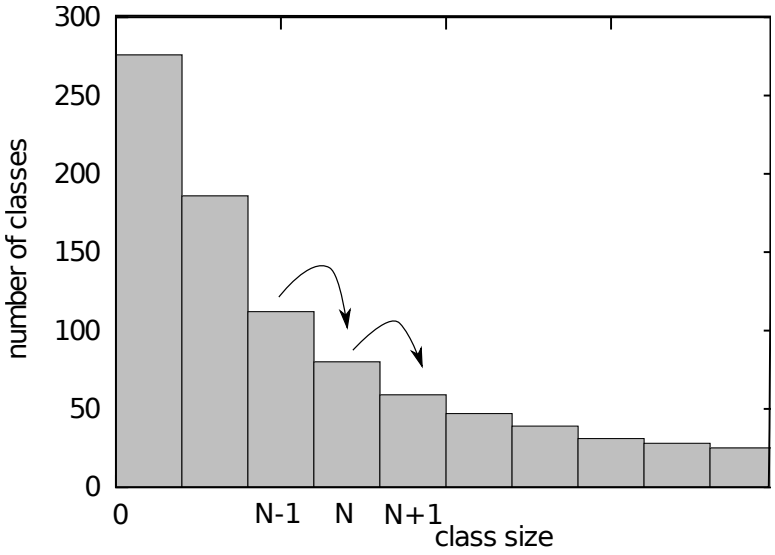


Figure 13: Illustration of equation 3.3.4. Individual classes grow constantly i.e., move to the right over time, as indicated by arrows. A stationary distribution means that the height of each bar remains constant.

3.3.2 *Preferential attachment models for networks and trees*

A similar model has been formulated for network growth by Barabási and Albert Barabási and Albert [1999], which explains the formation of a power law distribution in connectivity degree of nodes. It assumes that the networks grow in terms of nodes and edges, and that every newly added node to the system connects with a fixed number of edges to existing nodes. Attachment is again preferential, i.e. the probability for a newly added node i to connect to a certain existing node j is proportional to its number of existing edges of node j.

A node in the Barabási-Albert (BA) model corresponds to a class in Yule's model, and a new edge to a newly assigned element. Every added edge counts both to the degree of an existing node j, as well as to the newly added node i. It is always counted twice, so the existing nodes j and the newly added node i grow always by the same number of edges. This is why $m = 1$ and consequently $\beta = 2$ in the BA-model, *independently* of the number of edges that each new node creates.

This seminal model has been extended in many ways. For hierarchical taxonomies, we use a preferential attachment model for trees by Klemm et al. [2005]. The authors considered growth via directed edges, and explain power law formation in the *in-degree*, i.e. the edges directed from children to parent in a tree structure. In contrast to the BA-model, newly added nodes and existing nodes do not increase their in-degree by the same amount, since new nodes start with an in-degree of 0. Leaf nodes thus cannot attract attachment of nodes,

and preferential attachment alone cannot lead to a power-law. A small random term ensures that some nodes attach to existing ones independently of their degree, which is the analogous to the start of a new class in the Yule model. The probability v_i that a new node attaches as a child to the existing node i of with indegree d_i becomes

$$v_i = w\frac{d_i - 1}{D} + (1 - w)\frac{1}{D}, \tag{3.3.7}$$

where D is the size of the system measured in the total number of in-degrees. $w \in [0, 1]$ denotes the probability that the attachment is preferential, $(1 - w)$ the probability that it is random to any node, independently of their numbers of indegrees. As it has been done for the Yule process Simon [1955], Newman [2005a], Geipel et al. [2009], Tessone et al. [2011], the stationary distribution is again derived via the master equation (3.3.4). The exponent of the asymptotic power law in the in-degree distribution is $\beta = 1 + 1/w$. This model is suitable to explain scaling properties of the tree or network structure of large-scale web taxonomies, which have also been analyzed empirically, for instance for subcategories of Wikipedia Capocci et al. [2006].

3.3.3 *Model for hierarchical web taxonomies*

We now apply these models to large-scale web taxonomies like DMOZ. Empirically, we uncovered two scaling laws: (a) one for the size distribution of leaf categories and (b) one for the indegree (child-to-parent link) distribution of categories (shown in Figure 11). Since (a) and (b) arise jointly, we propose here a model generating the two scaling laws in a simple generic manner. A combination of the two processes detailed in subsections 3.3.1 and 3.3.2 may describe the growth process: websites are continuously added to the system, and classified into categories by human referees. At the same time, the categories are not a mere set, but form a tree structure, which grows itself in two quantities: in the number nodes (categories) and in the number of in-degrees of nodes (child-to-parent edges).

Based on the rules for voluntary referees of the DMOZ how to classify websites, we propose a simple combined description of the process. The database grows in *three* quantities:

(i) *Growth in websites.* New websites are assigned into category k, with probability $p(k) \propto N_k$ (Figure 14). This assignment happens independently of the hierarchy level of category k. However, only leaf categories may receive documents.

(ii) *Growth in categories.* With probability $1/m$, referees assign a website into a newly created category, at any level of the hierarchy (Figure 15). This

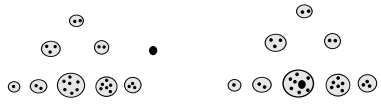


Figure 14: A website is assigned to existing categories with $p(k) \propto N_k$.

assumption would suffice to create a power law ignoring category size distribution, but since a tree-structure among categories exists, we also assume that the event of category creation is also attaching to the tree structure. The probability $v(d_i)$ that a category is created as the child of a certain parent category i can depend in addition on the *in-degree* d_i of that category.

Figure 15: (ii): Growth in categories is equivalent to growth of the tree structure in terms of in-degrees.

(iii) *Growth in children categories.* Finally, the hierarchy may also grow in terms of levels, since with a certain probability $(1 - w)$, new children categories are assigned independently of the number of children, i.e. its in-degree d_i of the category i. (Figure 16). Like in Klemm et al. [2005], the attachment probability to a parent i is therefore

$$v_i = w\frac{d_i - 1}{D} + (1 - w)\frac{\epsilon_i}{D}\,. \tag{3.3.8}$$

Equation (3.3.7) where $\epsilon_i = 1$ would suffice to explain power law in-degrees d_i and in category sizes N_i.

Figure 16: (iii): Growth in children categories.

To link the two processes more plausibly, it can be assumed that the second term in equation (3.3.8) denoting assignment of new 'first children' depends on the size N_i of parent categories,

$$\epsilon_i = \frac{N_i}{N}\,, \tag{3.3.9}$$

since this is closer to the rules by which the referees create new categories, but is not essential for the explanation of the power laws. It reflects that

the bigger a leaf category, the higher the probability that referees create a child category when assigning a new website it.

To summarize, the central idea of this joint model is to consider two measures for the size of a category: the number of its websites N_i (which governs the preferential attachment of new websites), and its in-degree, i.e. the number of its children d_i, which governs the preferential attachmet of new categories. To explain the power law in the category sizes, assumptions (i) and (ii) are the requirements. For the power law in the number of indegrees, assumptions (ii) and (iii) are the requirements. The empirically found exponents $\beta = 1.3$ and $\gamma = 1.9$ yield a frequency of new categories $1/m = 0.3$, and a frequency of new indegrees $(1 - w) = 0.9$.

3.3.4 *Other interpretations*

Instead of assuming in Equations (3.3.8) and (3.3.9) that referees decide to open a single child category, it is more realistic to assume that an existing category is *restructured*, i.e. one or several child categories are created, and websites are moved into these new categories such that the parent category contains less websites or even none at all. If one of the new children categories inherits all websites of the parent category (see Figure 17), the Yule model applies directly. If the websites are partitioned differently, the model contains effective shrinking of categories. This is not described by the Yule model, and the master Equation (3.3.4) considers only growing categories. However, it has been shown Tessone et al. [2011], Metzig and Gordon [2014] that also models including shrinking categories also lead to the formation of power laws. Further generalizations compatible with power law formation are that new categories do not necessarily start with one document, and that the frequency of new categories does not need to be constant.

Figure 17: Model without and with shrinking categories. In the left figure, a child category inherits all the elements of its parent and takes its place in the size distribution.

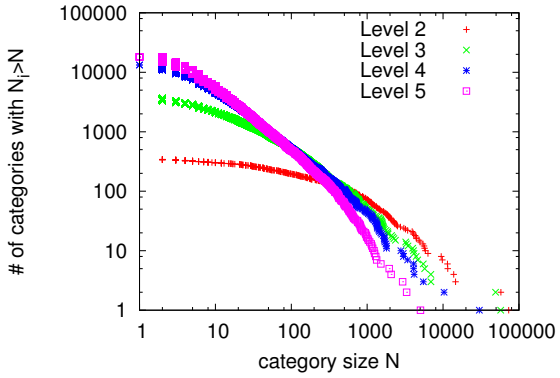


Figure 18: Category size distribution for each level of the LSHTC2-DMOZ dataset.

3.3.5 *Limitations*

However, Figures 10 and 11 do not exhibit perfect power law decay for several reasons. Firstly, the dataset is limited. Secondly, the hypothesis that assignment probability (3.3.2) depends uniquely on the size of a category might be too strong for web directories, in view of changing importance of topics. This may lead to big categories which receive only few new documents or none at all. In the work of Dorogovtsev and Mendes [2000], the authors have studied this problem by introducing an assignment probability that decays exponentially with age. For a low decay parameter they show that the stronger this decay, the steeper the power law; for strong decay, no power law forms. A last reason might be that referees re-structure categories in ways strongly deviating from the rules (i)- (iii).

3.3.6 *Statistics per level in the hierarchy*

The tree-structure of a database allows also to study the sizes of class belonging to a given level of the hierarchy. As shown in Figure 12 the DMOZ database contains 5 levels of different size. If only classes on a given level l of the hierarchy are considered, we equally found a power law in category size distribution as shown in Figure 18. Per-level power law decay has also been found for the in-degree distribution. This result may equally be explained by the model introduced above: Equations 3.3.2 and 3.3.8 respectively, are valid also if instead of $p(k)$ one considers the conditional probability $p(l)p(i|l)$, where $p(l) = \frac{\sum_{i'=1,l}^{\kappa} N_{i',l}}{\sum_{i'=1}^{\kappa} N_{i'}}$ is the probability of assignment to a given level, and $p(i|l) = \frac{N_{i,l}}{\sum_{i'=1,l}^{\kappa} N_{i',l}}$ the probability of being assigned to a given class within that

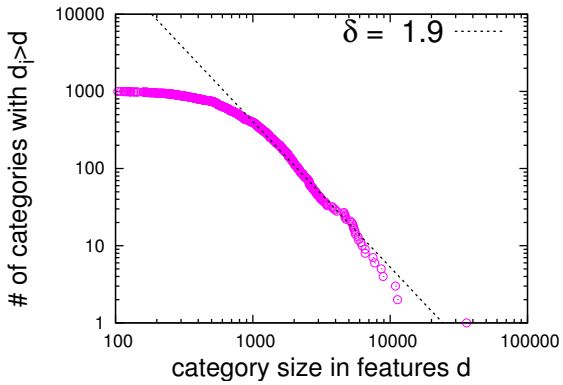


Figure 19: Number of features vs number of documents of each category.

level. The formation process may be seen as a Yule process *within* a level if $\sum_{i'=1,l}^{\kappa} N_{i',l}$ is used for the normalization in Equation 3.3.2, and this formation happens with probability $p(l)$ that a website gets assigned into level l. Thereby, the rate at m_l at which new classes are created need not be the same for every level, and therefore the exponent of the power law fit may vary from level to level. Power law decay for the per-level class size distribution is a straightforward corollary of the described formation process, and will be used in Section 5 to analyse the space complexity of hierarchical classifiers.

3.4 SPACE COMPLEXITY ANALYSIS

The fit of power law distribution to large-scale web taxonomies highlights the underlying structure and semantics which are useful to visualize important properties of the data especially in big data scenarios. In this section we focus on the applications in the context of large-scale hierarchical classification, wherein the fit of power law distribution to such taxonomies can be leveraged to concretely analyze the space complexity of large-scale hierarchical classifiers in the context of a generic linear classifier deployed in top-down hierarchical cascade.

3.4.1 *Relation between category size and number of features*

Having explained the formation of two scaling laws in the database, a third one has been found for the number of features in each category (see Figure 21). This is a consequence of the law in category sizes, shown in Figure 19. The result is closely related to the empirical Heaps' law Egghe [2007], stating that

the number of distinct words R in a document is related to the length n of a document as follows

$$R(n) = Kn^{\alpha} \tag{3.4.1}$$

where the empirical α is typically between 0.4 and 0.6. For the LSHTC2-large dataset, Figure 20 shows that for the collection of words and the collection of websites, similar exponents are found. An interpretation of this result is that the total number words in a category can approximately be measured by the number of websites in a category, although not all websites have the same length.

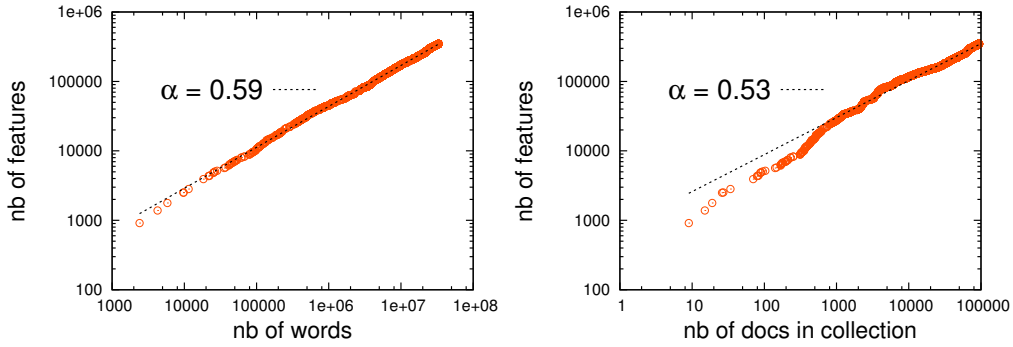


Figure 20: Heaps' law: number of distinct words vs. number of words, and vs number of documents.

Figure 20 (b) shows that bigger categories contain also more features, but this increase is weaker than the increase in websites. This implies that less very large categories exist, which is also reflected in the higher exponent $\delta = 1.9$ of a power-law fit in figure 19, (compared to the slower decay of the category size distribution in terms of number of documents shown in figure 10 where $\beta = 1.3$). Comparison of the exponents empirically yields that $\delta \cdot \alpha = 1.1$ which is lower than the empirical $\beta = 1.3$, but in the same order of magnitude.

In the following sections we first present formally the task of hierarchical classification and then we proceed to the space complexity analysis for large-scale systems. Finally, we empirically validate the derived bounds.

3.4.2 *Space Complexity of Large-Scale Classification*

The prediction speed for large-scale classification is crucial for its application in many scenarios of practical importance. It has been shown in Yang et al. [2003] that hierarchical classifiers have lower computational complexity of training as compared to flat classifiers. Furthermore, it has also been emphasized in the work of Bengio et al. [2010], Gao and Koller [2011] that prediction time can be logarithmic in the number of classes for top-down classification as compared

to flat classification. However, given the large physical memory of modern systems, what also matters in practice is the size of the trained model with respect to the available physical memory. To our knowledge, this aspect on space complexity of large-scale hierarchical classifiers has not been formally addressed so far. We, therefore, compare the space complexity of hierarchical and flat methods which governs the size of the trained model in large-scale classification. The goal of this analysis is to determine the conditions under which the size of the hierarchically trained linear model is lower than that of flat model.

For the space complexity in hierarchical classification, we use the notational setup as discussed in Section 2.2. 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. The hierarchical relationship among categories implies a transition from generalization to specialization as one traverses any path from root towards the leaves. This implies that the documents which are assigned to a particular leaf also belong to the inner nodes on the path from the root to that leaf node. In the case of flat classification, the hierarchy $\mathcal{H}$ is ignored, $\mathcal{Y} = V$, and the problem reduces to the classical supervised multiclass classification problem.

As a prototypical classifier, we use a linear classifier of the form $\mathbf{w}^T\mathbf{x}$ which can be obtained using standard algorithms such as Support Vector Machine or Logistic Regression. In this work, we apply one-vs-all $L2$-regularized $L2$-loss support vector classification as it has been shown to yield state-of-the-art performance in the context of large scale text classification Fan et al. [2008]. For flat classification one stores weight vectors $\mathbf{w}_y, \forall y$ and hence in a K class problem in d dimensional feature space, the space complexity for flat classification is:

$$Size_{Flat} = d \times K \tag{3.4.2}$$

which represents the size of the matrix consisting of K weight vectors, one for each class, spanning the entire input space.

We need a more sophisticated analysis for computing the space complexity for hierarchical classification. In this case, even though the total number of weight vectors is much more since these are computed for all the nodes in the tree and not only for the leaves as in flat classification. Despite this, the size of hierarchical model can be much smaller as compared to flat model in the large scale classification. The main insight behind this phenomenon is that when the feature set size is high (top levels in the hierarchy), the number of classes is less, and on the contrary, when the number of classes is high (at the bottom), the feature set size is low.

In order to analytically compare the relative sizes of hierarchical and flat models in the context of large scale classification, we assume power law behavior with respect to the number of features, across levels in the hierarchy. More precisely, if the categories at a level in the hierarchy are ordered with respect to the number of features, we observe a power law behavior. This has been validated from our analysis in the previous section 3.4.1 based on Heaps law and also been verified empirically as illustrated in Figure 21 for various levels in the hierarchy, for one of the datasets used in our experiments. More formally, the feature size $d_{l,r}$ of the r-th ranked category, according to the number of features, for level l, $1 \leq l \leq L-1$, is given by:

$$d_{l,r} \approx d_{l,1} r^{-\beta_l} \tag{3.4.3}$$

where $d_{l,1}$ represents the feature size of the category ranked 1 at level l and $\beta > 0$ is the parameter of the power law. Using this ranking as above, let $b_{l,r}$ represent the number of children of the r-th ranked category at level l ($b_{l,r}$ is the branching factor for this category), and let B_l represents the total number of categories at level l. Then the size of the entire hierarchical classification model is given by:

$$Size_{Hier} = \sum_{l=1}^{L-1} \sum_{r=1}^{B_l} b_{l,r} d_{l,r} \approx \sum_{l=1}^{L-1} \sum_{r=1}^{B_l} b_{l,r} d_{l,1} r^{-\beta_l} \tag{3.4.4}$$

Here level $l = 1$ corresponds to the root node, with $B_1 = 1$.

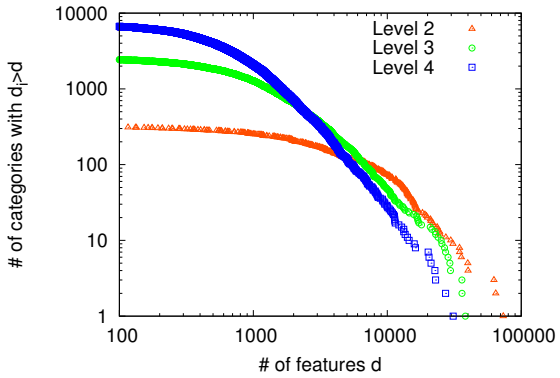


Figure 21: Power-law variation for features in different levels for LSHTC2-a dataset, Y-axis represents the feature set size plotted against rank of the categories on X-axis

We now state a proposition that shows that, under some conditions on the depth of the hierarchy, its number of leaves, its branching factors and power law parameters, the size of a hierarchical classifier is below that of its flat version.

Proposition 1. *For a hierarchy of categories of depth L and K leaves, let $\beta = \min_{1 \leq l \leq L} \beta_l$ and $b = \max_{l,r} b_{l,r}$. Denoting the space complexity of a hierarchical*

classification model by $Size_{hier}$ and the one of its corresponding flat version by $Size_{flat}$, one has:

$$For\ \beta > 1, if\ \beta > \frac{K}{K - b(L-1)} (> 1),\ then \quad Size_{hier} < Size_{flat} \tag{3.4.5}$$

$$For\ 0 < \beta < 1, if\ \frac{b^{(L-1)(1-\beta)} - 1}{(b^{(1-\beta)} - 1)} < \frac{1-\beta}{b} K, then \quad Size_{hier} < Size_{flat} \tag{3.4.6}$$

Proof. As $d_{l,1} \leq d_1$ and $B_l \leq b^{(l-1)}$ for $1 \leq l \leq L$, one has, from Equation 3.4.4 and the definitions of β and b:

$$Size_{hier} \leq bd_1 \sum_{l=1}^{L-1} \sum_{r=1}^{b^{(l-1)}} r^{-\beta}$$

One can then bound $\sum_{r=1}^{b^{(l-1)}} r^{-\beta}$ using (Yang et al. [2003]):

$$\sum_{r=1}^{b^{(l-1)}} r^{-\beta} < \left[\frac{b^{(l-1)(1-\beta)} - \beta}{1-\beta} \right] \text{ for } \beta \neq 0, 1 \tag{3.4.7}$$

leading to, for $\beta \neq 0, 1$:

$$\begin{aligned} Size_{hier} &< bd_1 \sum_{l=1}^{L-1} \left[\frac{b^{(l-1)(1-\beta)} - \beta}{1-\beta} \right] \\ &= bd_1 \left[\frac{b^{(L-1)(1-\beta)} - 1}{(b^{(1-\beta)} - 1)(1-\beta)} - (L-1) \frac{\beta}{(1-\beta)} \right] \end{aligned} \tag{3.4.8}$$

where the last equality is based on the sum of the first terms of the geometric series $(b^{(1-\beta)})^l$.

If $\beta > 1$, since $b > 1$, it implies that $\frac{b^{(L-1)(1-\beta)} - 1}{(b^{(1-\beta)} - 1)(1-\beta)} < 0$. Therefore, inequality (3.4.8) can be re-written as:

$$Size_{hier} < bd_1 (L-1) \frac{\beta}{(\beta - 1)}$$

Using our notation, the size of the corresponding flat classifier is: $Size_{flat} = Kd_1$, where K denotes the number of leaves. Thus:

$$\text{If } \beta > \frac{K}{K - b(L-1)} (> 1), \text{then } Size_{hier} < Size_{flat}$$

which proves Condition (3.4.5).

The proof for Condition (3.4.6) is similar: assuming $0 < \beta < 1$, it is this time the second term in Equation 3.4.8 ($-(L-1)\frac{\beta}{(1-\beta)}$) which is negative, so that one obtains:

$$Size_{hier} < bd_1 \left[\frac{b^{(L-1)(1-\beta)} - 1}{(b^{(1-\beta)} - 1)(1-\beta)} \right]$$

and then:

$$\text{If } \frac{b^{(L-1)(1-\beta)} - 1}{(b^{(1-\beta)} - 1)} < \frac{1-\beta}{b} K, \text{then } Size_{hier} < Size_{flat}$$

which concludes the proof of the proposition. □

It can be shown that condition 3.4.6 is satisfied for a range of values of $\beta \in]0, 1[$. However, as is shown in the experimental part, it is condition 3.4.5 of Proposition 1 that holds in practice. In order to empirically validate the claim of Proposition 1, we measured the trained model sizes of a standard top-down hierarchical scheme (TD), which uses a linear classifier at each parent of the hierarchy, and the flat one.

We use the publicly available DMOZ data of the LSHTC challenge which is a subset of Directory Mozilla. More specifically, we used the large dataset of the LSHTC-2010 edition and two datasets were extracted from the LSHTC-2011 edition. These are referred to as LSHTC1-large, LSHTC2-a and LSHTC2-b respectively in Table 11. The fourth dataset (IPC) comes from the patent collection released by World Intellectual Property Organization. The datasets are in the LibSVM format, which have been preprocessed by stemming and stopword removal. Various properties of interest for the datasets are shown in Table 11. Table 4 shows the difference in trained model size (actual value of the model size on the hard drive) between the two classification schemes for the four datasets, along with the values defined in Proposition 1. The symbol $\bigtriangledown$ refers to the quantity $\frac{K}{K-b(L-1)}$ of condition 3.4.5.

Dataset	#Training Inst.	#Test Inst.	#Classes	#Feat.	Tree Depth
LSHTC1-large	93,805	34,880	12,294	347,255	6
LSHTC2-a	25,310	6,441	1,789	145,859	6
LSHTC2-b	36,834	9,605	3,672	145,354	6
IPC	46,324	28,926	451	1,123,497	4

Table 3: Datasets for hierarchical classification with the properties: Number of training/test examples, target classes and size of the feature space.

Dataset	$Size_{hier}$	$Size_{Flat}$	β	b	$\bigtriangledown$
LSHTC1-large	2.8	90.0	1.62	344	1.12
LSHTC2-a	0.46	5.4	1.35	55	1.14
LSHTC2-b	1.1	11.9	1.53	77	1.09
IPC	3.6	10.5	2.03	34	1.17

Table 4: Model size (in GB) for flat and hierarchical models along with the corresponding values defined in Proposition 1. The symbol $\bigtriangledown$ refers to the quantity $\frac{K}{K-b(L-1)}$

As shown for the three DMOZ datasets, the trained model for flat classifiers can be an order of magnitude larger than for hierarchical classification. This results from the sparse and high-dimensional nature of the problem which is quite typical in text classification. For flat classifiers, the entire feature set participates for all the classes, but for top-down classification, the number of classes and features participating in classifier training are inversely related, when traversing the tree from the root towards the leaves. As shown in Proposition 1, the power law exponent β plays a crucial role in reducing the model size of hierarchical classifier.

The previous proposition complements the analysis presented in Yang et al. [2003] in which it is shown that the training and test time of hierarchical classifiers is importantly decreased with respect to the ones of their flat counterpart. In this work we show that the space complexity of hierarchical classifiers is also better, under a condition that holds in practice, than the one of their flat counterparts. Therefore, for large scale taxonomies whose feature size distribution exhibit power law decay, hierarchical classifiers should be better in terms of speed than flat ones, due to the following reasons:

1. As shown above, the space complexity of hierarchical classifier is lower than flat classifiers.
2. For K classes, only $O(\log K)$ classifiers need to be evaluated per test document as against $O(K)$ classifiers in flat classification.

3.5 CONCLUSION

In this work we presented a model in order to explain the dynamics that exist in the creation and evolution of large-scale taxonomies such as the DMOZ directory, where the categories are organized in a hierarchical manner. More specifically, the presented process jointly models the growth in the size of the

categories (in terms of documents) as well as the growth of the taxonomy in terms of categories, which to our knowledge have not been addressed in a joint framework. From this, we derive with the help of Heaps's law a third scaling law in the features size distribution of categories which we then exploit for performing an analysis of the space complexity of linear classifiers in large-scale taxonomies. We provided a quantitative analysis of the space complexity for hierarchical and flat classifiers and proved that the complexity of the former is always lower than that of the latter. The analysis has been empirically validated in several large-scale datasets from publicly available web-taxonomies. The space complexity analysis can be used in order to estimate beforehand the size of trained models for large-scale data. This is of importance in large-scale systems where the size of the trained models may impact the inference time.

4

EXPLOITING DATA-DISTRIBUTION FOR LEARNING

Using the power-law distribution of data among categories in large-scale category systems, we study two algorithms which aim at achieving (a) better classification accuracy, and (b) efficient model-selection leading to faster training. The fit to power-law distribution implies that a significant fraction of categories, referred to as *rare categories*, have very few documents assigned to them. For large-scale datasets which exhibit this property, it leads to the following two challenges, (i) categories with extremely few training documents in them make it harder for learning algorithms to learn effective decision boundaries which can correctly detect such categories in the test set, and (ii) computational complexity of hyper-parameter tuning for learning algorithms such as SVM by the commonly used k-fold cross-validation is extremely high. We present techniques which exploit the power-law distribution of documents among categories to address these challenges. More concretely, (i) we propose a soft-thresholding based framework for classification which leads to better classification in the presence of rare categories and secondly, (ii) we present a computationally efficient model selection in large-scale classification. Finally, in the context of large-scale hierarchical classification, we propose a method which effectively combines discriminative and generative classifiers by leveraging the variation in the number of training examples to the number of features at nodes in the root to leaf path in the hierarchy. The classifier ensemble leads to faster training and prediction, without sacrificing significantly on classification accuracy. The empirical evaluation on publicly available large-scale datasets from the LSHTC challenge demonstrate that the proposed methods address effectively the challenges of better classification accuracy and lower computational complexity.

4.1 SOFT-THRESHOLDING FOR CLASSIFICATION IN POWER-LAW DISTRIBUTED CATEGORIES

Due to the tremendous growth in data from various sources such as social networks, web-directories and digital encyclopedias, big data analytics and large scale learning have gained increasing importance in recent years and have become a key focus of academia and industry alike. Directory Mozilla, for instance, lists over 5 million websites distributed among close to 1 million categories. Another commonly used instances of large-scale encyclopedias and category systems include Wikipedia and Medical Subject Heading hierarchy of the National Library of Medicine is another instance of a large-scale classification system in the domain of life sciences In order to minimize the amount of human effort involved in maintaining such large-scale scenarios, there is a definite need to automate the process of classifying data into the target categories. However, as studied in the previous chapters that most large-scale category systems exhibit fit to power-law distribution. This attribute of large-scale datasets poses major research challenge for building good classification systems.

4.1.1 *Power-law distribution*

As discussed in Chapter 3, and also shown empirically in the work by Yang et al. [2003], Liu et al. [2005] that the distribution of documents among categories in large category systems exhibits a fit to power-law distribution. Formally, let N_r denote the size of the r-th ranked category (in terms of number of documents), then :

$$N_r = N_1 r^{-\beta} \tag{4.1.1}$$

where N_1 represents the size of the 1-st ranked category and $\beta > 0$ denotes the exponent of the power law distribution. The fat-tailed power law distribution highlights the fact that many categories have very few documents assigned to them. For instance, 76% of the categories in the Yahoo! directory have less than 5 documents in them Gopal and Yang [2013b].

Due to the fat-tailed power law distribution, a large number of categories have very few documents assigned to them. It is, therefore, statistically harder to learn good decision boundaries for these categories. The decision boundaries of the bigger categories are more *attractive* as compared to the rare categories. As a result, a test instance which actually belongs to one of the rare categories is assigned to a bigger category. On one hand, this leads to high False Positive rate for bigger categories, and on the other hand, rare categories are lost in the

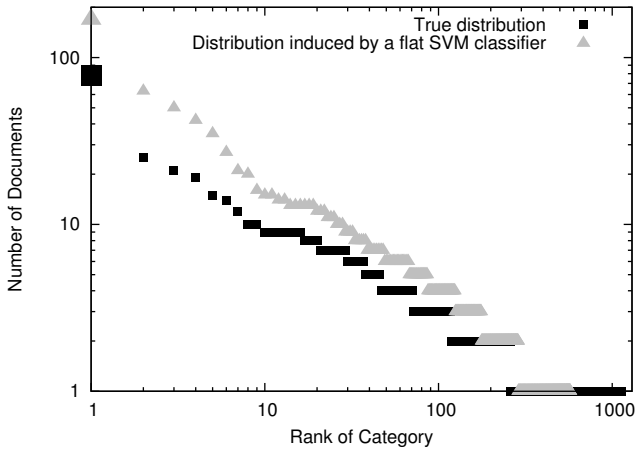


Figure 22: Comparison of distribution of test instances among categories in the true distribution and in the distribution induced by a flat SVM classifier; the X-axis represents the rank of categories (by number of documents) and Y-axis the number of documents in those categories. Categories with same number of documents effectively have same rank.

classification process. This is shown for one of the LSHTC datasets in Figure 22, which depicts

1. The true distribution of test instances among target categories denoted by grey triangles, and
2. The distribution of documents among categories induced when a flat (multi-class) SVM classifier is used for classification, denoted by solid black squares.

For the distribution induced by the SVM classifier, observations in Figure 22 which demonstrate the high False-positive rate for large categories and inability to detect rare categories in such distributions are :

- On the left side of the plot, the graph for the distribution induced by the SVM classifier starts higher and remains higher as compared to true distribution, but drops much sharply on the right part, and
- Comparing the tails of the distributions on the right side of the plot, the true distribution has a fatter tail as compared to the induced distribution, i.e., it has many more categories of 1 or 2 documents as compared to the distribution induced by the SVM classifier.

More concretely, the category with the maximum number of documents in the true distribution has 78 documents (denoted by bigger solid square in black on Y-axis), while in the induced distribution it has 176 documents (denoted by bigger solid triangle in grey on Y-axis). Furthermore, the actual number of

categories in the test distribution is 1139, while the flat SVM classifier is able to detect merely 574 categories.

4.1.2 *Related work and our contributions*

Not only limited to flat SVM classifier, the state-of-the-art methods such as Gopal and Yang [2013b] also suffer from these two problems mentioned which is also apparent in low values of the Macro-F1 measure achieved by these methods. The work by Liu et al. [2005] is among the pioneering studies in classification of power-law distributed web-scale directories such as the Yahoo! directory consisting of over 100,000 target classes. For similar category systems, classification techniques based on *refined experts* and *deep classification* have been proposed in Bennett and Nguyen [2009] and Xue et al. [2008] respectively. More recently recursive regularization based SVM (HR-SVM) has been studied in Gopal and Yang [2013b] wherein the optimization problem for learning the discriminant functions exploits the given taxonomy of categories. This approach represents the current state-of-art as it performs better than most techniques on large-scale datasets released as part of the Large Scale Hierarchical Text Classification Challenge in last few years [1]. Other studies related to large-scale learning are presented in works such as Perronnin et al. [2012], Bengio et al. [2010], Gao and Koller [2011], Deng et al. [2011]. However, the above studies do not focus on the specific problem of rare-category detection in large-scale power-law distributed category systems, which is the focus of this section.

To address the problem of rare-category detection in large-scale power-law distributed category systems, we propose an easy to implement method which performs post-processing on the posterior probabilities of categories given the instance. More concretely, we proceed as follows, (i) we propose a simple but useful upper bound on the accuracy of any classifier which classifies documents into target categories and hence induces a distribution of documents among them, and (ii) we then present a soft-thresholding based algorithm which aims to increase the value of the bound bound on the accuracy derived in the first step and thereby favoring rare categories. This scheme performs better than the state-of-the-art HR-SVM technique in both Micro-F1 and Macro-F1 measures, and especially for the latter, at a much lower computational complexity. Also, the relative improvement in the total number of categories detected in classification is as high as 20% on some datasets.

1 http://lshtc.iit.demokritos.gr/

4.1.3 *Accuracy Bound on Power-law Distributed Categories*

Now we propose an upper bound on the accuracy of a given classifier C. Unlike most learning theoretic error bounds Vapnik [1998], Mohri et al. [2012], the nature of this bound is quite simple and is particularly suited for classification problems with a large number of target categories. The derivation of the upper bound on the accuracy of the classifier C is based on the distribution of unseen instances induced by it among the target categories.

We consider mono-label multi-class classification problems, where observations $\mathbf{x}$ lie in an input space $\mathcal{X} \subset \mathbb{R}^d$ and belong to one and only one category from a discrete set $\mathcal{Y}$ of labels, where $|\mathcal{Y}| > 2$. We suppose that examples are pairs of $(\mathbf{x}, y)$, with $y \in \mathcal{Y}$, identically and independently distributed (i.i.d) according to a fixed, but unknown probability distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$. We further assume to have access to a training set $S_{train} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{m}$ also generated i.i.d with respect to $\mathcal{D}$. In the context of text classification, $\mathbf{x}^{(i)} \in \mathcal{X}$ denotes the vector representation of document i and its label $y^{(i)} \in \mathcal{Y}$ represents the category associated with $\mathbf{x}^{(i)}$. Using the statistics of the training data, we first provide confidence intervals for the estimate of the prior probability for each category.

Lemma 1. *Let m denote the total number of instances in the training set such that the category y_ℓ consists of m_ℓ instances. Let p_{y_ℓ} denote the true prior probability for category $y_\ell \in \mathcal{Y}$ and $\frac{m_\ell}{m} \triangleq \hat{p}_{y_\ell}$ its empirical estimate. Then $\forall \delta$, such that $0 < \delta \leq 1$, with probability at least $(1 - \delta)$, the following upper bound holds simultaneously for all categories,*

$$\forall y_\ell \in \mathcal{Y}, p_{y_\ell} \leq \hat{p}_{y_\ell} + \sqrt{\frac{\log |\mathcal{Y}| + \log \frac{1}{\delta}}{2m}} \tag{4.1.2}$$

where the probability is computed with respect to repeated samples of the training set.

The above lemma can be proved by applying Hoeffding's inequality and then union bound for it to hold simultaneously for all $|\mathcal{Y}|$ categories.

Proof. Using Hoeffding's inequality for random variables bounded in the interval $[0, 1]$, we have

$$\forall \epsilon > 0, Pr(p_{y_\ell} - \hat{p}_{y_\ell} > \epsilon) \leq exp\left(-\frac{2m^2\epsilon^2}{m}\right) = \frac{\delta}{|\mathcal{Y}|}$$

where $Pr(e)$ represents the probability of event e. Solving for the deviation ϵ in terms of δ gives the required inequality on the right hand side. It can similarly be proved for the inequality on the left hand side. The $\log |\mathcal{Y}|$ factor in the bound is a result of fact that the bound should hold simultaneously for all $|\mathcal{Y}|$ categories. □

Using the bound in inequality (4.2.2), we now present a probabilistic upper bound on the accuracy of a classifier C evaluated on an independent set S which is also generated i.i.d. from $\mathcal{D}$.

Theorem 1. *Let $S = \{(\boldsymbol{x}^{(j)}, y^{(j)})\}_{j=1}^{n}$ be a set generated i.i.d. from $\mathcal{D}$. Let n_ℓ^C be the number of examples in S assigned to category y_ℓ by the classifier C which is trained on S_{train}. Then for any $0 < \delta \leq 1$, the following bound on the accuracy of C over S, denoted by $Acc(C)$, holds with probability at least $(1 - \delta)$:*

$$Acc(C) \leq \frac{1}{|S|} \sum_{\ell=1}^{|\mathcal{Y}|} \min\{(\hat{p}_{y_\ell} \times |S|), n_\ell^C\} \triangleq B(Acc(C)) \tag{4.1.3}$$

where $\hat{p}_{y_\ell}$ denotes the estimate on the prior probability of the category y_ℓ in the training set as computed in Lemma 2.

Proof. For $\ell = 1$, $(\hat{p}_{y_1} \times |S|)$ represents a probabilistic upper bound on the number of instances in category y_1 and using Lemma 2, the bound holds with probability $(1 - \delta/|\mathcal{Y}|)$, where $|S|$ denotes the size of S. Clearly, the maximum number of instances for category y_1 that can be correctly classified by C is given by $\min\{(\hat{p}_{y_1} \times |S|), n_\ell^C\}$. Summing over all $|\mathcal{Y}|$ categories gives an *upper bound* on the total number of instances that can possibly be correctly classified by C with confidence atleast $(1 - \delta)$. The maximum accuracy rate of classifier C is, therefore, given by right hand side of (4.1.3). □

Even though the bound given in Equation (4.1.3) seems loose, it is indeed quite useful when learning classifiers on a large number of target categories which are power-law distributed. For the dataset used in Figure 22, the actual accuracy of the flat classifier is 0.45 and the upper bound as given by Equation 4.1.3 is 0.64. In the next section, we propose a ranking-based algorithm which aims at improving this upper bound. Intuitively, for a given test instance, instead of predicting the top-ranked category in terms of posterior probabilities, our algorithm performs a soft-thresholding by ranking them and then post-processing the result by majority voting to encourage highly ranked rare categories. As shown in our experiments that the resulting method not only leads to higher value of the upper bound (0.71 for the dataset used in Figure 22) but also tends to have higher values of both Micro-F1(= accuracy) and Macro-F1 measure.

4.1.4 *Soft-thresholding Algorithm for Higher Bound-value*

The $\min(.,.)$ function in the bound derived in equation (4.1.3) has two arguments, where the first argument corresponds to the estimate of the number of instances in category ℓ and the second argument is the number of instances

assigned to this category by the classifier C. As a result, a higher value of the bound is achieved for C, if the two arguments are close to each other for large number of categories. On the other hand, if C assigns a large number of false-positives to categories which have large number of training instances in them, the value attained by the bound will be lower since :

1. For most of the large categories, the first argument in $\min(.,.)$ will be accounted towards computing the bound. This is due to the fact that these categories will attract many false-positives from small categories and hence making the second argument of $\min(.,.)$ bigger.
2. For a large fraction of the small categories which have false-negatives, the second argument in $\min(.,.)$ will be close to zero and will be used in the computation of the bound.

The two problems correspond to the left and right portions respectively in Figure 22 for the distribution induced by the flat SVM classifier. As also shown in our experiments, the bound on the accuracy as given by equation (4.1.3) also captures the variation in the true accuracy and hence can be used as its proxy. Therefore, when dealing with large number of target classes the bound on the accuracy represents a criterion which can be improved in order to obtain better classification. It may be noted that the bound represents a necessary condition for a classifier C to have high accuracy. It does not provide a sufficient condition since it is possible in an adversarial setup to achieve an upper bound of 1 by simply assigning the test instances to categories in the same proportion as in the training set.

With the aim of having a higher value of the accuracy bound, (in equation (4.1.3)) by reducing the False positive rate for top-ranked categories and detecting more of the rare categories, we present an efficient algorithm which achieves better measures for Micro-F1 and Macro-F1. Given the training set S_{train}, we first train a multi-class SVM (using Liblinear) which can give probabilistic output. When predicting the category associated to the test instance $\mathbf{x}$, the algorithm first computes the class posterior probabilities $(\hat{p}_{y_l}|\mathbf{x}), \forall 1 \leq l \leq |\mathcal{Y}|$ and ranks the categories according to posterior probabilities. Let $y_{r1} = \arg\max_{y_l \in \mathcal{Y}} (\hat{p}_{y_l}|\mathbf{x})$ be the first-ranked category and $y_{r2} = \arg\max_{y_l \in \{\mathcal{Y} - y_{r1}\}} (\hat{p}_{y_l}|\mathbf{x})$ is the second-ranked category. Also, let $m_{y_{r1}}$ and $m_{y_{r2}}$ be the number of training instances in these categories in the training set S_{train}. For the instance $\mathbf{x}$, we define a predicate $pred(\mathbf{x})$ which is true if and only if the following conditions are satisfied :

1. the difference $(\hat{p}_{y_{r1}}|\mathbf{x}) - (\hat{p}_{y_{r2}}|\mathbf{x}) \leq \Delta$, and
2. $m_{y_{r1}}/m_{y_{r2}} \geq R$.

If $pred(\mathbf{x})$ evaluates to true, it implies that $\mathbf{x}$ may be wrongly classified by the flat SVM classifier to category y_{r1}. In this scenario, a majority-voting based

re-prediction to *distinguish the top two categories* for $\mathbf{x}$ is performed as follows. An *instantaneous training set* is created by randomly under-sampling the top-ranked category to match the number of training instances in the rare category, and all the training instances from the rare category are used. Using this instantaneous training set, a binary classifier is then trained and the class of the instance $\mathbf{x}$ is re-predicted. The above process of creation of instantaneous set, training and prediction is repeated an odd number of times and one of the categories from $\{y_{r1}, y_{r2}\}$ with majority votes is finally predicted. This post-processing of the output is performed for a small fraction of the instances in the test set for which $pred(\mathbf{x})$ evaluates to true. Moreover, since it involves only top-two categories, it adds only marginal computational cost as compared to learning the multi-class SVM for all the categories. The proposed soft-thresholding based re-ranking procedure is given below in Algorithm 1:

Algorithm 1 Proposed Algorithm

Input: Training data S_{train} and Test data S_{test}
Output: Labels for S_{test}
  Learn Multiclass SVM (Crammer-Singer algorithm Crammer and Singer [2002])
  for each test instance $\mathbf{x} \in S_{test}$ **do**
    Predict posterior probabilities $(\hat{p}_{y_l}|\mathbf{x}), \forall 1 \leq l \leq |\mathcal{Y}|$
    if $pred(\mathbf{x})$ is true **then**
      Create *instantaneous training set* t (odd) times
      To distinguish $\{y_{r1}, y_{r2}\}$, learn t binary classifiers
      Re-predict instance $\mathbf{x}$ with each binary classifier
      Output from $\{y_{r1}, y_{r2}\}$ the one with majority votes
    else
      Output category $\arg\max_{y_l \in \mathcal{Y}}(\hat{p}_{y_l}|\mathbf{x})$
    end if
  end for
  return Labels $\forall \mathbf{x} \in S_{test}$

As shown in our experiments, re-ranking the class posterior probabilities based on this algorithm yields significant improvement in the Macro-F1 and Micro-F1 measures as compared to state-of-art methods. The parameters Δ and R used in Algorithm 1 are chosen by cross-validation and we observed that even intuitive values such as $R = 5$ and $\Delta = 1/(10 \times |\mathcal{Y}|)$ give comparable results as compared to state-of-the-art HR-SVM method. It may also be noted that the proposed algorithm can be extended to consider top-k categories instead of top-2, which is one of our future works.

Is it similar to handling class-imbalance? It may be noted that the nature of class imbalance problem posed in the large-scale datasets with thousands of

power-law distributed categories is different from the traditional classification problems in low-dimensional space such as in UCI datasets. A typical rare category in large-scale category systems consists of 2-to-4 instances and spans a very low dimensional sub-space of a few hundreds of features in the entire feature space which could be as big as hundreds of thousand dimensions, as shown in Table 11. This is in contrast to conventional imbalanced data-sets which lie in feature spaces of few tens of dimensions and all classes span the entire dimensionality of the entire feature space. As a result, the conventional methods of handling class-imbalance such as class-wise penalty in SVM (which penalizes a mis-classification for a class inversely in the ratio of number of instances in that class) do not improve classification in such settings. We tested this technique on our datasets and the results were poorer as compared to normal class-insensitive penalization. We therefore did not pursue this strategy any further.

4.1.5 *Experimental Evaluation*

Dataset	Training/Test instances	Categories $\lvert\mathcal{Y}\rvert$	Features d
LSHTC-2010-s	4,463/1858	1,139	51,033
LSHTC-2010-l	128,710/34,880	12,294	381,580
LSHTC-2012	383,408/103,435	11,947	348,548

Table 5: LSHTC datasets and their properties

We present empirical results on publicly available Directory Mozilla (DMOZ) datasets from the LSHTC challenge in 2010 (s and l suffixes correspond to smaller and larger versions) and 2012. The statistics of the data are shown in Table 11. The number of features, denoted by d, represents the number of distinct words in the vocabulary after stemming and stop-word removal. The datasets are in the LibSVM format with term-frequency information for each document.

The metrics used for comparison are Micro-F1 measure and Macro-F1 measure, which are computed as follows:

- **Micro-F1** : It is an instance based evaluation measure and weighs higher those categories which have higher fraction in the test set. Let TP_y, FP_y and FN_y denote respectively the true-positives, false-positives, and false-

negatives for the class label $y \in \mathcal{Y}$. Then Micro-F1 measure is given by

$$P = \frac{\sum_{y \in \mathcal{Y}} TP_y}{\sum_{y \in \mathcal{Y}} TP_y + FP_y}$$

$$R = \frac{\sum_{y \in \mathcal{Y}} TP_y}{\sum_{y \in \mathcal{Y}} TP_y + FN_y}$$

$$Micro - F1 = \frac{2PR}{P + R}$$

- **Macro-F1** : It is a category-based evaluation measure and weighs all categories equally and hence is more sensitive to the ability of the classifier to detect rare-categories. It is given by

$$P_y = \frac{TP_y}{TP_y + FP_y}$$

$$R_y = \frac{TP_y}{TP_y + FN_y}$$

$$Macro - F1 = \frac{1}{|\mathcal{Y}|} \sum y \in \mathcal{Y} \frac{2P_y R_y}{P_y + R_y}$$

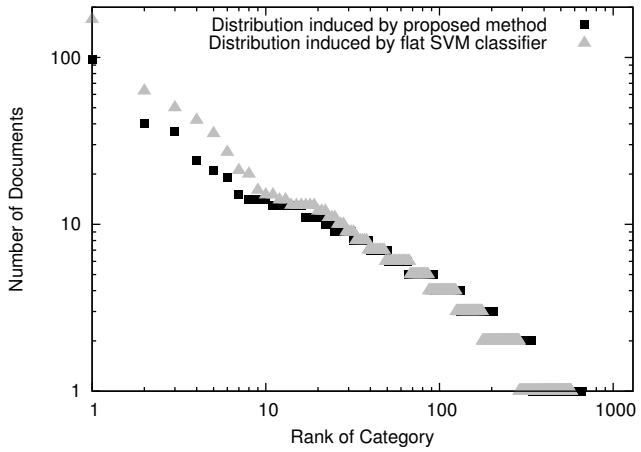


Figure 23: Comparison of distribution of test instances among categories for the method proposed in Algorithm 1 and SVM baseline. X-axis representing the rank (by number of documents) of categories and Y-axis the number of documents in them.

The parameters Δ and R used in Algorithm 1 are chosen by cross-validation and we observed that even intuitive values work well in practice. In Table 6, we compare the algorithm proposed in Section 4.1.4 with HR-SVM from the recent work in Gopal and Yang [2013b] and also against the SVM-baseline.

Dataset	Algorithm 1	HR-SVM Gopal and Yang [2013b]	CS-SVM
LSHTC-2010-s			
Micro-F1	**47.36††**	45.31	45.15
Macro-F1	**32.91††**	28.94	29.40
B(Acc(C))	**0.71**	0.63	0.64
Categories detected	**658**	570	574
Training Time	**1.1x**	1.7x	1x
LSHTC-2010-l			
Micro-F1	**46.67††**	46.02	45.82
Macro-F1	**34.65††**	33.12	32.63
B(Acc(C))	**0.77**	0.73	0.72
Categories detected	**8523**	8102	8039
Training Time	**1.1x**	1.6x	1x
LSHTC-2012			
Micro-F1	**57.78††**	57.17	56.44
Macro-F1	**34.15††**	33.05	31.59
B(Acc(C))	**0.76**	0.72	0.70
Categories detected	**8220**	7965	7882
Training Time	**1.1x**	1.6x	1x

Table 6: Comparison of Micro-F1 and Macro-F1 for the proposed algorithm, HR-SVM and CS-SVM (Crammer-Singer). The training time is shown as a multiple of time taken by the SVM-baseline. The variation of the bound value derived in Equation 4.1.3 and the number of categories detected by each method is also shown. The significance-test results (using Micro sign test for Micro-F1 measure and using Macro t-test for the Macro-F1 measures, as proposed in Yang et al. [2003]) are denoted for a p-value less than 1%.

Comparison of the approaches shows that the proposed method, aimed at improving the value of the accuracy bound (4.2.2) yields improvement over the state-of-the-art HR-SVM technique. The results of the significance test are shown with respect to HR-SVM Gopal and Yang [2013b] and SVM-baseline, and †† represents significant improvement over both the methods. Since our method is explicitly targeted at rare category detection, the improvement in Macro-F1 measure is particularly significant, which confirms that the method is able to correctly recognize rare categories. For instance, the relative improvement in Macro-F1 over HR-SVM for LSHTC-2010-s dataset is close to 15%. This is also

confirmed by the comparison of the number of detected categories for each of the three methods. For the **LSHTC-2010-s** dataset, the relative increase in the number of detected categories is almost as high as close to 20%.

Figure 23 shows the distribution of test instances induced by the method proposed in Algorithm 1 for the **LSHTC-2010-s** dataset. On comparing Figure 23 with Figure 22, we observe that the distribution induced by our method is much closer to the true distribution as compared to the flat SVM classifier. Two important observations follow from the comparison:

- The left part of the plot shows that bigger categories have a lower False positive rate as compared to SVM classifier.
- The tail of the distribution shows that our method detects more rare categories, which further confirms better rate of Macro-F1 measure as compared to state-of-art methods.

To compare the computational cost of each method, training times are also shown in Table 6. The comparison to HR-SVM shows that our method enjoys favorable performance in terms of computational complexity. Since Algorithm 1 uses flat baseline as a first step, and re-training is performed only for a fraction of test instances (in on-line fashion), its cumulative training time is slightly more than that for flat-baseline.

4.1.6 *Remarks*

In this section, we focused on the specific problem of rare-category detection in large-scale power-law distributed category systems. However, for classification in large-scale category systems consisting of tens of thousand classes in few hundred thousand dimensional feature spaces, we are still faced with many computational bottlenecks. One them being the computational complexity of k-fold cross-validation for hyper-parameter selection such the λ parameter in SVM training. For instance, on one of the LSHTC datasets consisting of 0.5 million training documents among 36,000 categories, 5-fold cross-validation to learn the parameter λ will take around one month on a single quad-core machine with standard hardware. In the next section, we discuss in detail on this research challenge and also propose a computationally-efficient method for hyper-parameter tuning in the context of power-law distributed categories.

4.2 EFFICIENT MODEL-SELECTION IN BIG DATA

In the first part of this chapter, we proposed a method to deal with the skewness of data in large-scale category systems from the classification accuracy viewpoint. Another challenge posed by such datasets is the sheer scale of the classification task and hence, the scalability of typically used classification algorithms such as Support Vector Machines and Logistic Regression. Since the number of target classes is of the order of tens of thousands and feature set size corresponding to the vocabulary is of the order of hundreds of thousands, it is also computationally expensive to learn such discriminative classifiers. As also discussed in the recent work by Gopal and Yang [2013a], the LSHTC-large dataset having 12,294 categories in a feature set of size 347,256 one needs to learn $12,294 \times 347,256 = 4.2$ billion parameters. In such scenarios, model-selection techniques such as k-fold cross-validation to tune the regularization parameter λ of SVM classifier for 7 values $\{10^{-3}, 10^{-2}, \ldots, 10^{2}, 10^{3}\}$ for $k = 5$ would require the process of learning 4.2 billion parameters 35 times. As another instance, on the Wikipedia-2011 dataset used in our experiments which has approximately 0.5 million training documents among 36,000 categories, 5-fold cross-validation to learn the parameter λ will take around one month on a single quad-core machine with standard hardware.

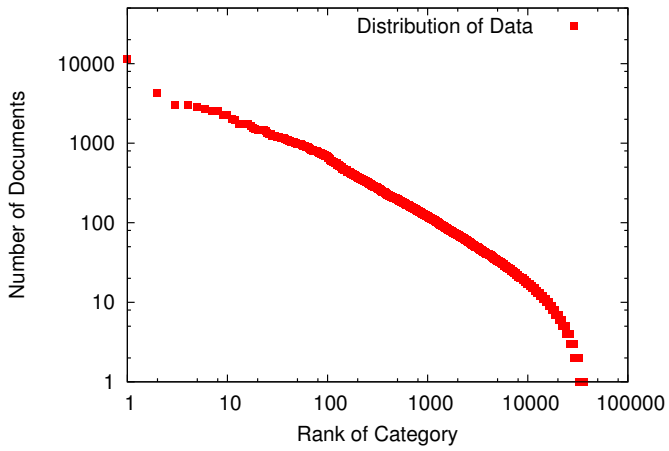


Figure 24: Distribution of 456,866 training instances (for a Wikipedia subset) among 36,000 categories in the training data, with X-axis representing the rank (by number of documents) of categories and Y-axis the number of documents in those categories. Some of the characteristics of the data : (i) The categories with maximum number of documents in the training distribution has 11,400 documents, (ii) Approximately 15,000 of the 36,000 categories have ≤ 5 documents, with 4,000 categories having just 1 document in the training set.

Due to the presence of a large number of *rare categories* in power-law distributed category systems, training on a fraction of the given data (for better computational efficiency) is also not desirable. This method ignores useful information especially for such categories and hence leads to a sub-optimal choice of the hyper-parameter λ, as is also verified in our experiments. As a result, model selection for classification in large-scale web directories suffer from two major challenges:

- Using the entire data makes the process of model selection (such as k-fold cross-validation) **computationally expensive**,
- Using a fraction of data for computational efficiency leads to **sub-optimal parameter choice**.

Therefore, conventional techniques in machine learning offer no promising alternative to computationally expensive k-fold cross-validation Mohri et al. [2012] for large-scale web directories. The large-scale nature of the problem, coupled with the scarcity of sufficient number of training instances for the *rare categories* poses a research and engineering challenge in order to design scalable systems with good prediction performance.

In this work, to address the issues of computational complexity of model selection, we propose an efficient alternative to cross-validation. Specifically, our contributions are the following: (i) We show that the accuracy bound developed in the first part of this chapter naturally motivates an efficient scheme for hyper-parameter tuning, and (ii) we demonstrate empirically that by employing the proposed technique, one can speed-up the hyper-parameter search by a factor of k as compared to k-fold cross-validation.

4.2.1 *Related Work*

The work by Liu et al. [2005] is among the pioneering studies for classification of web-scale directories such as the Yahoo! directory consisting of thousands of target categories. The authors study the distribution of documents among categories and verify the fit to power-law distribution in such taxonomies. They apply this phenomena to analyze the performance with respect to accuracy and training time complexity for flat and hierarchical classification. Other techniques have been recently proposed for classification in large-scale settings such Bennett and Nguyen [2009], Xue et al. [2008], Gopal et al. [2012].

The HR-SVM based technique proposed in Gopal and Yang [2013b] represents the current state of art for most of the bench-mark datasets. However, this relies on computationally expensive cross-validation to search for the appropriate value of the regualrization parameter λ in the range $\{10^{-3}, 10^{-2}, \ldots, 10^{2}, 10^{3}\}$. Therefore, for classification problems involving tens of thousand of target

categories and training documents in the range of hundreds of thousand, this mandates the use of high performance and parallel-processing based computing systems such Hadoop. Even though *k*-fold cross-validation is easily parallelizable, our method can also exploit a parallel computation infrastructure and is more efficient in this set-up as well. We would also like to note that efficiency in model selection by exploring the regularization path of SVM has been studied in Hastie et al. [2004], Friedman et al. [2010]. However, the aim in those works is to be able to perform cross-validation in finite number of points instead of $\mathbb{R}_+$. Though related on a high-level, the focus of contribution in these works is quite different to the problem addressed in our work.

4.2.2 *Accuracy Bound for Classification in Large Number of Categories*

In this section, we recall from the previous section, the upper bound on the accuracy of a given classifier C. In our experiments, we show that this bound serves as a good proxy for the actual accuracy of C, and further exploit this intuition to perform model selection.

Using the problem setup from the first part of this chapter, we recall the first result from the previous section wherein we present confidence interval on estimate of the prior probability for each category in a large-scale category system.

Lemma 2. *Let m denote the total number of instances in the training set such that the category y_ℓ consists of m_ℓ instances. Let p_{y_ℓ} denote the true prior probability for category $y_\ell \in \mathcal{Y}$ and $\frac{m_\ell}{m} \triangleq \hat{p}_{y_\ell}$ its empirical estimate. Then $\forall \delta$, such that $0 < \delta \leq 1$, with probability at least $(1-\delta)$, the following upper bound holds simultaneously for all categories,*

$$\forall y_\ell \in \mathcal{Y}, p_{y_\ell} \leq \hat{p}_{y_\ell} + \sqrt{\frac{\log |\mathcal{Y}| + \log \frac{1}{\delta}}{2m}} \tag{4.2.1}$$

where the probability is computed with respect to repeated samples of the training set.

Using this inequality, we re-call the result from the previous section which presents a probabilistic upper bound on the accuracy of a classifier C. The goal of a classification algorithm, such as a Support Vector Machine, is to learn a classifier C which maximizes the accuracy on the test set S_{test}.

Theorem 2. *Let $S_{test} = \{(\boldsymbol{x}^{(j)}, y^{(j)})\}_{j=1}^{n}$ be the test set which is also generated i.i.d. from $\mathcal{D}$. Let n_ℓ^C be the number of examples in S_{test} assigned to category y_ℓ by the*

classifier C which is trained on S_{train}. Then for any $0 < \delta \leq 1$, the following bound on the accuracy of C over S_{test}, denoted by $Acc(C)$, holds with probability at least $(1 - \delta)$:

$$Acc(C) \leq \frac{1}{|S|} \sum_{\ell=1}^{|\mathcal{Y}|} \min\{(\hat{p}_{y_\ell} \times |S_{test}|), n_\ell^C\} \triangleq B(Acc(C)) \tag{4.2.2}$$

where $\hat{p}_{y_\ell}$ denotes the estimate on the prior probability of the category y_ℓ in the training set as computed in Lemma 2.

Equation (4.2.2) shows that a classifier C is likely to have higher value of the bound provided n_k^C is close to $(\hat{p}_{y_k} \times |S_{test}|), \forall k$. On the other hand, a classifier which assigns a large number of false-positives to large classes due to imbalanced nature of the problem will be penalized because of the following two reasons :

1. The bound involves $min(.,.)$ and for a large class k with lots of false-positives, the first term in $min(.,.)$ will be accounted towards the computation of the bound, and
2. For small classes which have false-negatives, the second term in $min(.,.)$ will be close to zero and will be used in the computation of the bound.

As also shown in our experiments, the bound on the accuracy as given by equation (4.2.2) captures the variation in the true accuracy. Therefore, when dealing with large number of target classes the bound on the accuracy can be viewed as a proxy for the test set accuracy.

4.2.3 *Using accuracy bound as alternative to k-fold cross-validation*

Training process of effective learning algorithms such as SVM or Logistic Regression requires learning billions of paramters for web-scale datasets. These discriminative learning algorithms minimize a combination of empirical error and model complexity. The template of the objective function which is minimized is of the following form:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} R_{emp}(\mathbf{w}) + \lambda \; Reg(\mathbf{w}) \tag{4.2.3}$$

where $Reg(\mathbf{w})$ is the regularization term to avoid complex models and $R_{emp}(.)$ represents the empirical error. For SVM classifier, $R_{emp}(.)$ is based on hinge-loss $(\max(0, 1 - y_i \mathbf{w}^T \mathbf{x}_i))$ and for Regularized Logistic Regression $R_{emp}(.)$ is based on logistic loss $(\log(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)))$. The hyper-parameter λ controls the trade-off between the empirical error and regularization term.

Algorithm 2 demonstrates model selection via k-fold cross-validation for learning the hyper-parameter λ. The inner for-loop requires the computationally expensive process to be repeated k times for each value of the hyper-parameter.

Algorithm 2 Model selection using k-fold cross-validation

Require: Training data $S_{tr} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{m}$, learning algorithm such as SVM
  Randomly permute the training data instances
  Split S_{tr} into k parts
  for each value of $\lambda \in \{10^{-3}, 10^{-2}, \ldots, 10^{2}, 10^{3}\}$ **do**
    for $(l = 1; l \leq k; l++)$ **do**
      Train on all parts except the l-th part
      Test on the l-th part and compute accuracy acc_l^{λ}
    end for
    Compute average accuracy$(= \frac{1}{k}\sum_{l=1}^{k} acc_l^{\lambda})$ for current value of λ
  end for
  Return the value of λ with highest accuracy

Algorithm 3 Model selection using accuracy bound (4.2.2)

Require: Training data $S_{tr} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{m}$, $S_{test} = \{(\mathbf{x}^{(j)}, y^{(j)})\}_{j=1}^{n}$ and learning algorithm such as SVM
  Randomly permute the training data instances
  for each value of $\lambda \in \{10^{-3}, 10^{-2}, \ldots, 10^{2}, 10^{3}\}$ **do**
    Train an SVM model using S_{tr}
    Test the model on S_{test}
    Compute the accuracy bound (4.2.2) for each value of λ
  end for
  Return the value of λ with highest accuracy bound

Algorithm 3 presents an alternative to k-fold cross-validation based on the accuracy bound derived in equation (4.2.2) that can instead be employed for tuning the hyper-parameter λ. For a given learning algorithm (such as SVM), different settings of the hyper-parameter (λ) are likely to yield a different classifier (separating hyper-plane). Tuning the hyper-parameter is, therefore, reduced to the problem of finding a classifier which yields highest value of the bound in equation (4.2.2) on the test set.

The advantage of this strategy is that it avoids the need to repeat the process k times and hence its computational cost is same as 1-fold cross-validation. As shown in our experiments, this strategy for model selection works well and chooses the same value as found by k-fold cross-validation. For each value of λ, it computes the upper bound derived in equation (4.2.2) and selects the value with the highest value for the bound.

k-fold cross-validation on fraction of training data In large-scale scenarios, one common alternative to speed-up model selection process such as k-fold cross-validation is to use a fraction of data instead of using the entire data. We

Dataset	#Training/#Test instances	#Categories	#Features	#Parameters
DMOZ-2010-s	4,463/1,858	1,139	51,033	58,126,587
DMOZ-2011	36,834/9,605	3,672	145,354	533,739,888
DMOZ-2010-l	128,710/34,880	12,294	381,580	4,691,144,520
Wiki-2011	456,866/81,262	36,504	346,299	12,641,298,696
IPC	46,324/28,926	451	1,123,497	506,697,147

Table 7: Datasets used, along with their properties: Number of training instances, test instances, target categories, size of the feature space and number of parameters learnt. Each of the DMOZ datasets and IPC dataset has 1 label per training/test instance, while the Wikipedia dataset has 1.85 labels on average for the training set.

employ linear[2] SVM in our experiments and the computational complexity of linear SVM is linear in number of training instances. Therefore, one can only select (1/k) fraction of training data such that the computational complexity of training an SVM using *k*-fold cross-validation (Algorithm 2) is same for the proposed method (Algorithm 3).

However, this leads to sub-optimal choice of the hyper-parameter as was observed in our experiments. This is primarily due to the fact that all datasets (except one) exhibit fit to power-law distribution as shown in Figure 24 such that most categories have few documents assigned to them. For instance, on the Wikipedia dataset approximately 40% of the categories have less than 6 documents in them. As a result, using a small fraction of the training data makes the task of learning a good classifier even more difficult for such rare categories. Therefore, using a fraction of the training data for computational efficiency is undeseriable in large-scale datasets with large number of categories.

4.2.4 *Experimental Evaluation*

Dataset Description

We used several publicly available datasets to empirically verify the applicability of the bound derived in equation (4.2.2) as a an efficient alternative to *k*-fold cross-validation. The datasets used for our experiments are the following:

- DMOZ-subsets which are derived from Directory Mozilla and are available from the 2010 and 2011 editions of the LSHTC challenge.

2 In large-scale and high dimensional data, as in document classification, which is almost linearly separable, computationally efficient linear SVM performs at par with kernel versions.

- IPC dataset which corresponds to patent categorization from International Patent Classification [3]
- Wikipedia which is derived from Wikipedia and also available from the 2011 edition of the LSHTC challenge.

The important statistics of the datasets (such as the sizes of the training/test sets, feature set, number of target categories and number of parameters to be learnt) are shown in Table 11. For instance, the smallest dataset (DMOZ-2010-small) considered in our experiments has approximately 58 million parameters and the largest one (Wikipedia-2011) has approximately 12 billion parameters.

The DMOZ datasets are a subset of the Directory Mozilla and are single-labled datasets, i.e. each training/test instance is associated to a single target category. The Wikipedia subset which is much bigger in size is multi-labeled with average labels per instance in the training set being 1.85. For the multi-labeled Wikipedia dataset, we trained one binary SVM for each class. In order to select the number of labels for each test instance we used the meta-labeler approach which learns a regression model that predicts the number of labels Tang et al. [2009a]. For each test instance the decisions of the binary SVMs are ordered according to their confidence and we keep the first k' labels, where k' is the number of labels that is predicted by the meta-labeler model.

The IPC dataset is also a single labeled dataset which consists of relatively fewer target categories as compared to DMOZ and Wikipedia datsets. Another difference of the IPC dataset (as compared to DMOZ and Wikipedia dataset) is that it does not exhibit fit to power-law distribution.

Methods Compared In order to empirically measure the effectiveness of the proposed method for efficient model selection in large-scale classification problem, we present two sets of comparisons:

- **Proposed Algorithm 3 vs k-fold cross-validation** : We first verify the ability of our method proposed to find the same hyper-parameter as by k-fold cross-validation.
- **Using entire training data vs $\frac{1}{k}$ fraction** : We also empirically verify the effectiveness of k-fold cross-validation on $\frac{1}{k}$ of the training data.

The classifier applied (on all but the multi-label Wikipedia dataset) is the Multi-class SVM as proposed in Crammer and Singer [2002] and implemented in Liblinear package Fan et al. [2008]. The hyper-parameter considered is the trade-off parameter λ between the empirical error (measured by ξ_i's) and multi-

3 http://www.wipo.int/classifications/ipc/en/ITsupport/Categorization/dataset/wipo-alpha-readme.html

class margin (measured by $\mathbf{w}_k$'s) in following optimization problem which has the form of the template equation (4.2.3):

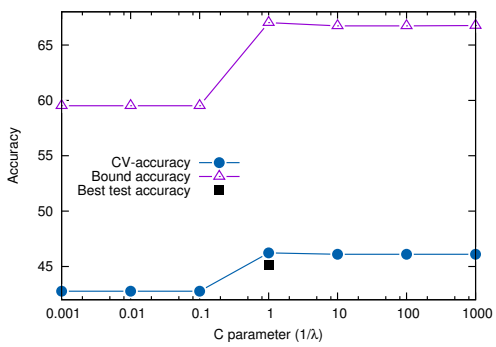
$$\min_{\mathbf{w}_k, \boldsymbol{\xi}} \sum_{i=1}^{m} \xi_i + \frac{\lambda}{2} \sum_{k=1}^{K} \mathbf{w}_k^T \mathbf{w}_k$$

subject to

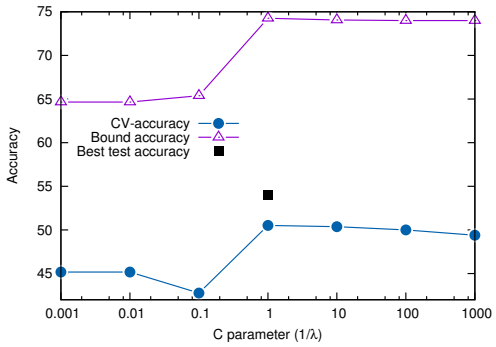
$$\mathbf{w}_{y_i}^T \mathbf{x}_i - \mathbf{w}_k^T \mathbf{x}_i \geq e_i^k - \xi_i, \ \forall i = 1, \ldots, M \text{ and } \xi_i \geq 0$$

where

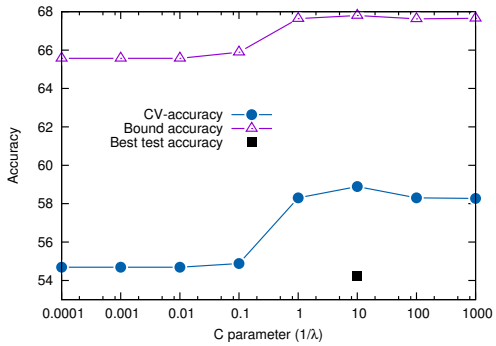
$$e_i^k = \begin{cases} 0 & \text{if } y_i = k \\ 1 & \text{if } y_i \neq k \end{cases}$$



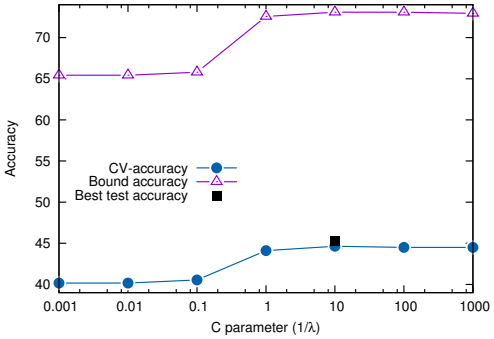
(a) DMOZ-2010-s



(b) DMOZ-2011



(c) IPC



(d) DMOZ-2011-l

Figure 25: Variation (with λ) of cross-validation accuracy (CV-accuracy), accuracy bound derived in equation 4.2.2 on the DMOZ-2010-small, DMOZ-2011-subset, IPC and DMOZ-2010-small datasets.The value of λ which attains the best test-set accuracy along with the corresponding accuracy value is displayed by the solid-black square.

4.2.5 *Results*

Algorithm 3 vs k-fold cross-validation Figure 25 shows the variation of cross-validation accuracy and the bound derived in equation (4.2.2) with the variation in λ in the range $\{10^{-3}, 10^{-2}, \ldots, 10^{2}, 10^{3}\}$ for the DMOZ and IPC datasets. The effectiveness of Algorithm 3 based on the bound derived in equation (4.2.2) for hyper-parameter tuning is demonstrated by the following two observations:

- The extent of variation in the bound with the change in λ mimics the variation in cross-validation accuracy. This suggests that this accuracy bound serves as a reliable proxy to measure the degree of variation in cross-validation accuracy.
- The hyper-parameter value which maximizes the bound and cross-validation accuracy is same for all datasets. Moreover, it also coincides with the hyper-parameter value which maximizes the test-set accuracy. This is shown in the solid-square dot in each of the sub-figures.

For the Wikipedia dataset, it was not possible to perform k-fold cross-validation for all values of the hyper-parameter in the range $\{10^{-3}, 10^{-2}, \ldots, 10^{2}, 10^{3}\}$, and hence only two values (10^{-2} and 10^{2}) were chosen on this dataset. Therefore, for large-scale datasets it may be computationally infeasible to perform cross-validation without using parallel computing infrastructure such as Hadoop. However, it may be noted that if a parallel computing platform is available, the proposed algorithm (Algorithm 3) can also benefit from it. The for-loop in the algorithm can be easily parallelized and the bound can be simultaneously computed for all settings of the hyper-parameter.

k-fold cross-validation on entire data vs $\frac{1}{k}$ fraction Table 8 presents the best parameter value selected by the cross-validation method that uses all the training data and the one that uses only $1/5$ of the data (denotes by $CV_{1/5}$) across all datasets. For each value, we also report the corresponding accuracy in the test set.

Dataset	λ_{CV}.	$\lambda_{CV_{1/5}}$	Accuracy(CV)	Accuracy ($CV_{1/5}$)
DMOZ-2010-s	**1.0**	10.0	**45.15**	44.94
DMOZ-2011	**1.0**	10.0	**54.01**	53.84
DMOZ-2010-l	**10.0**	10.0	**45.26**	44.17
IPC	**10.0**	1.0	**54.22**	53.59

Table 8: Parameter values and corresponding accuracy on the test set for λ obtained from cross-validation using the entire training data and its variation using $1/5$ of the available data ($CV_{1/5}$). With bold typeface the best parameter values and accuracies.

From the results it is clear that when reducing the available data in order to reduce the computation cost the model selection method makes sub-optimal decisions. In most of the cases, using the $CV_{1/5}$ method was unable to select the best parameter value. As reducing the training data the method is more biased but also the variance increases making more difficult the estimation of the performance. Even though the $CV_{1/5}$ has the same complexity as our method it leads to sub-optimal model selection and thus to inferior performance.

4.2.6 *Remarks*

In this section, we have highlighted the computational issues of k-fold cross-validation in large-scale datasets which are power-law distributed. The datasets such as Directory Mozilla are large-scale as well as consist of a large fraction of rare categories. We proposed an efficient alternative method to k-fold cross-validation for hyper-parameter selection in these scenarios, wherenin the proposed method exploits the side-information as given by the proposed bound. This can be seen as an instance of general paradigm of extracting latent information in Big Data to tackle the bottle-necks such as computational complexity of learning.

4.3 DATA-DEPENDENT CLASSIFIER SELECTION

With an increasing amount of data from various sources such as web advertizing, social media and images, automatic classification of unseen data to one of tens of thousand target classes has caught the attention of the research community. In flat classification, no relationship is assumed between the target classes and K classifiers are learnt, one for each of the K classes. If some semantic structure exists among the classes, such as hierarchical, as in a rooted tree (Figure 26), a multi-class classifier is trained on each of the non-leaf node in the tree to distinguish between each of its children. For large-scale classification, hierarchical strategies have two main advantages over flat classification:

- To classify a test instance, one needs to evaluate only $O(\log(K))$ classifiers, as against $O(K)$ for flat classification, and
- As shown in Chapter 5, hierarchical classification may lead to better (in general comparable) predictive performance as compared to flat techniques Liu et al. [2005], Babbar et al. [2013a]

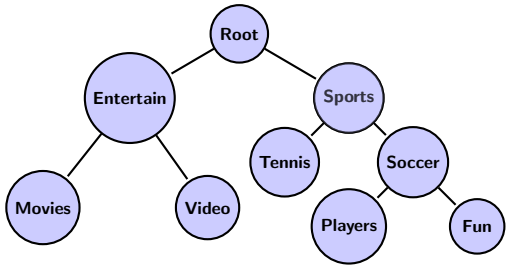


Figure 26: Sample Taxonomy of Classes

In the context of large-scale hierarchical classification (LSHC), open challenges like the Pascal Large Scale Hierarchical Text Classification (LSHTC) [4] and Imagenet Large Scale Visual Recognition Challenge (ILSVRC) [5] have been organized. In LSHTC for instance, the classes from the DMOZ and Wikipedia taxonomies are arranged in a rooted tree and directed acyclic graph respectively. The taxonomy thereby implicitly defines the semantic relationship among the classes. The publicly available DMOZ dataset, from the LSHTC challenge, contains around 400,000 training documents from the 27,875 target classes on the leaf nodes of the hierarchy tree with an extremely sparse representation involving 594,158 features. Outside of the LSHTC, various other approaches have also been proposed for large scale hierarchical classification, which have met with varying degrees of success (e.g., Bennett and Nguyen [2009], Xue et al. [2008], Gopal et al. [2012], Gopal and Yang [2013b]).

In terms of classification accuracy, discriminative learning algorithms such as SVM and Logistic Regression (LR) are known to learn better classifiers as compared to generative learning algorithms such as Naive Bayes. For this reason, discriminative classifiers have been on the fore-front when dealing with classification, as can be found in the works of Bengio et al. [2010], Perronnin et al. [2012], Cai and Hofmann [2004], Gopal and Yang [2013b]. Given training set consisting of a set of $(\mathbf{x}, y)$ pairs, unlike generative classifiers which model the joint probability $p(\mathbf{x}, y)$ and then use Bayes rule to compute $p(y|\mathbf{x})$, discriminative classifiers model the posterior $p(y|\mathbf{x})$, directly.

Discriminative versus Generative Classifiers As has been mentioned in the seminal work of Vapnik Vapnik [1998] about the choice of discriminative over generative classifiers, "one should solve the classification problem directly and never solve a more general problem as an intermediate step". This is in-line with our earlier observation on usage of discriminative classifiers such as SVM and LR over generative classifiers. On the other hand, Naive Bayes, which is one of most widely used generative classifier, has the following advantages over discriminative classifiers:

4 http://lshtc.iit.demokritos.gr/

5 http://www.image-net.org/challenges/LSVRC/2011/

- It has faster training time since learning the classifier amounts to counting occurrence of a word in training set. This is unlike SVM and LR classifiers, which require solving high dimensional optimization problems and hence have much higher computational complexity.
- In large-scale category systems since most words occur only in a small fraction of categories, the probability of a word occurring in a class takes default values for most $\langle word, class \rangle$ pairs. As a result, one can store the model for Naive Bayes classifier in an extremely sparse format which further leads lower space complexity and hence faster prediction time.

Therefore, in the context of large-scale taxonomies such as DMOZ, there is a tradeoff between prediction accuracy and computational complexity of training and prediction. In this part of the chapter, we study the tradeoffs between using generative models such as multinomial Naive Bayes, on one hand, and discriminative models such as Support Vector Machines (SVM) or Logistic Regression, on the other hand.

Furthermore, in the work on the theoretical properties relating to the sample complexity of Naive Bayes classifier as done in Ng and Jordan [2001], it has been shown that it can perform comparable or better than LR when the number of training instances is sub-linear (such as logarithmic) in the number of features. This implies that there are regimes of operation under which Naive Bayes classifier may be preferable as compared to discriminative classifiers. Therefore, under such circumstances, one can instead deploy Naive Bayes to get faster training and prediction speed without loosing on classification accuracy. In this part of the chapter, we discuss the variation of ratio of training sample size to the feature set size from the root of hierarchy towards the leaves. The variation in this ratio represents a difference in the regime which suits discriminative and generative classifiers differently. Therefore, to build an overall classification scheme, it is imperative to use classifiers which suit that particular local regime of operation. This leads to an ensemble of discriminative and generative classifiers deployed in a top-down hierarchical cascade useful scenario in which one could combine both types of models in the larger hierarchy to get the best of both worlds. An illustration of such a scheme of combining classifiers is shown in Figure 27 wherein on the left only SVM classifier is deployed and on the right a combination of SVM and Naives Bayes classifier is deployed.

In the light of the theoretical insight given in Ng and Jordan [2001], we now study the data distribution in large-scale taxonomies which determine choice of deploying discriminative (SVM) or generative classifier (NB) at various nodes in the top-down cascade.

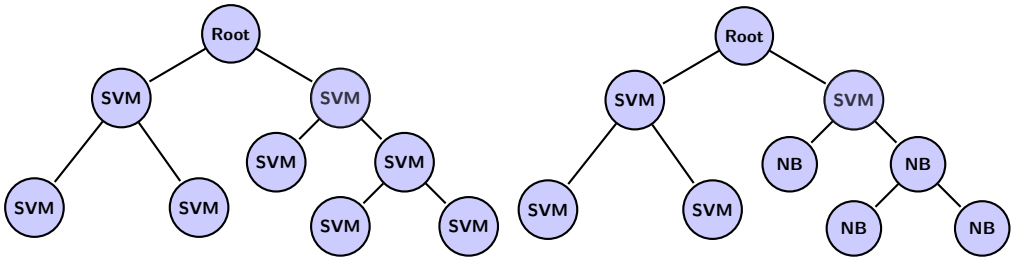


Figure 27: Top-down deployment of classifiers in uniform and hybrid fashion

4.3.1 *Sample Complexity and LSHC*

For hierarchical classification, 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 sisters $\mathfrak{S}(v) = \{v' \in V \setminus \{\perp\}; v \neq v' \wedge \pi(v) = \pi(v')\}$ and its daughters $\mathfrak{D}(v) = \{v' \in V \setminus \{\perp\}; \pi(v') = v\}$. 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

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

Given a new test instance $\mathbf{x}$, the goal is to predict the class $\widehat{y}$. We consider a top-down deployment of classifiers 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 until a leaf node is reached. At each non-leaf node $v \in V$, a score $f_c(\mathbf{x}) \in \mathbb{R}$ is computed for each daughter $c \in \mathfrak{D}(v)$ and the child $\widehat{c}$ with the maximum score is predicted i.e. $\widehat{c} = \arg\max_{c:(v,c)\in E} f_c(\mathbf{x})$. 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.

For our analysis, we focus on linear SVM and Multinomial Naive Bayes (NB) representing discriminative and generative models respectively. In SVM, $f_c(\mathbf{x})$ is modeled as a linear classifier such that $f_c(\mathbf{x}) = \mathbf{w}_c^T \mathbf{x}$. To learn a one-versus-

rest L2-regularized, L2-loss SVM-based discriminative classifier for node v, we solve the following optimization problem for each daughter c of v

$$\min_{\mathbf{w}_c, \xi} \frac{\lambda}{2}||\mathbf{w}_c||^2 + \sum_{i=1}^{m_v} \xi_{(i,c)}^2$$

The indices i above are such that $\forall i, 1 \leq i \leq m_v, y_i \in L_v$, were L_v denotes the set of leaves in the subtree rooted at node v and m_v denotes the number of training examples for which the root-to-leaf path passes through the node v. Furthermore, if $y_i \in L_c$ and $(v, c) \in E$, then the constraints for the above optimization problem are given by, $\forall i$

$$\mathbf{w}_c^t \mathbf{x}_i \geq 1 - \xi_{(i,c)}, \text{ and } \xi_{(i,c)} \geq 0$$

For the standard NB model in which predicted class is the one with maximum posterior probability, i.e.

$$\widehat{c} = \arg \max_{c:(v,c)\in E} \Pr(c|\mathbf{x}), \text{ s.t. } \Pr(c|\mathbf{x}) \propto \Pr(c)\Pr(\mathbf{x}|c)$$

and the probabilities are replaced by their maximum likelihood estimates, taking Laplace smoothing into account.

Classical Results on sample complexity

With SVM and Naive Bayes as defined described as our representative discriminative and generative classifiers, we now present relevant results from statistical learning theory Vapnik [1998] which deal with the sample complexity of these learning algorithms.

Proposition 2. *Vapnik [1998] For a binary classification problem in d-dimensional feature space with m training examples, let f_G and f_D represent the classifiers learnt by fitting generative and discriminative model respectively. Further, let $f_{G,\infty}$ and $f_{D,\infty}$ denote their corresponding asymptotic versions i.e. functions learnt when the sample size approaches infinity. Let $\varepsilon(.)$ be the function representing the generalization error of its argument, then these results can be summarized as follows :*

1. $\varepsilon(f_{D,\infty}) \leq \varepsilon(f_{G,\infty})$;
2. $\varepsilon(f_D) \leq \varepsilon(f_{D,\infty}) + O\left(\sqrt{\frac{d}{m}\log\frac{m}{d}}\right)$ *holds with high probability over random samplings of the m-sized training set.*

Proposition 3. *Ng and Jordan [2001] For a binary classification problem in d-dimensional feature space with m training examples, let f_G represents the classifier learnt by fitting generative and $f_{G,\infty}$ denotes its asymptotic version. Let $\varepsilon(.)$ be the function representing the generalization error of its argument, then with high probability :*

$$\varepsilon(f_G) \leq \varepsilon(f_{G,\infty}) + G\left(O\left(\sqrt{\frac{1}{m}\log n}\right)\right)$$

where $G(\tau)$ is upper bounded by $Pr_{\mathbf{x}}[l_{G,\infty}(\mathbf{x}) \in [-d\tau, d\tau]]$ and $l_{G,\infty}(.)$ represents the discriminant function corresponding to the decision function $f_{G,\infty}(.)$.

These two above results provide us with the following insights:

1. The asymptotic generalization error of discriminative classifier is smaller than that of a generative classifier,
2. Under finite training set sizes, in order to achieve the same generalization error as under asymptotic regime, discriminative classifier requires training instances which is atleast linear in the number of features, and
3. Under finite training set sizes, in order to achieve the same generalization error as under asymptotic regime, generative classifier requires training instances which is atleast logarithmic in the number of features

Taking into account these important theoretical insights, we study the variation of ratio of number of features to number of training examples at the different classification problem from the root to the leaves of the tree-based taxonomy as shown in Figure 27.

Data heterogeneity in large-scale taxonomies
For a multi-class classification problem at node v of the hierarchy, let d_v denote the dimensionality of the feature space and m_v denote the number of training documents for which the root-to-leaf path goes through node v. Let their ratio for node v be denoted by r_v, i.e. $r_v = \frac{d_v}{m_v}$.

In the context of large scale hierarchical classification, such as DMOZ, there is a wide spectrum over which r_v varies. For the classification problem corresponding to a node v at the top levels of the hierarchy tree, the ratio r_v is much higher as compared to its value for nodes at lower levels. Figure 28 shows the variation of average value of r_v for DMOZ dataset when plotted against the hierarchy levels. Each piece-wise linear curve in the plot corresponds to the class size range of the multi-class problem. Two important properties of the dataset, one of which follows from Figure 28, are: (i) The ratio r_v increases towards the leaves, and (ii) Almost 97% of the multi-class problems involve 2-15 classes. This shows that the nature of the learning problem posed is *different* in different parts of the hierarchy tree.

As a consequence of the above arguments, this implies the following design choices to build component classifiers for large scale hierarchical classification. We also briefly mention our observation for each of them in case of DMOZ data:

- On the nodes which are close to the root (including the root itself), we are close to the regime of asymptotic operation. Therefore using argument (1) from above, one should deploy discriminative classifiers such as SVM or logistic regression.

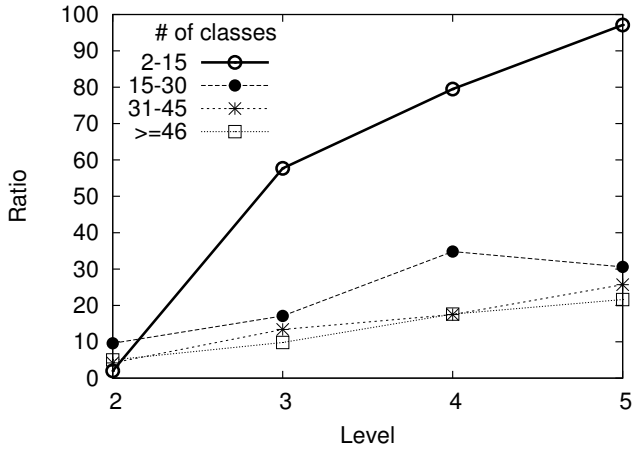


Figure 28: Variation in ratio of feature set size to training sample size with the hierarchy level. Level 2 corresponds to the children of root node and level 5 to the level that leads to leaves.

Observation for DMOZ : As shown in Figure 30, for level 1 and 2, SVM does indeed performs better and achieves much higher accuracy than NB classifier.

- Argument (3) above suggests that one should deploy NB classifier for the sub-problems lower down the hierarchy since for *most* of the nodes, m is upper bounded by $\lg(d)$ i.e. $m = O(\lg(d))$.
  Observation for DMOZ : As shown in Figure 30, for levels 4 and 5, NB cannot surpass the accuracy of SVM in this regime, which could be the result of argument (1). Importantly, however, the accuracy gap between the two classifiers is much smaller in this regime.

This indicates that, for lower levels in large hierarchy, NB is competitive to SVM and one can still employ NB instead of SVM, provided it can excel on metrics other than accuracy. In the next section, we discuss the deployment of an ensemble of NB and SVM classifiers in the top-down hierarchy tree.

Adaptive Classifier Selection

From the above observations for the DMOZ dataset, in order to perform well on the various measure of interest i.e., including (i) prediction accuracy, (ii) training time to train the classifiers, (iii) compact model size, and (iv) faster prediction speed, one therefore combine NB and SVM classifier in top-down cascade. This is illustrated in Figure 29 for a tree-based taxonomy with SVM classifier at the top-levels and NB classifier at the bottom levels. Since the NB classifier is faster to train and leads to more compact models, one can load all the classifiers of the hierarchy in the physical memory and can get massive speedup for prediction, without sacrificing on prediction accuracy.

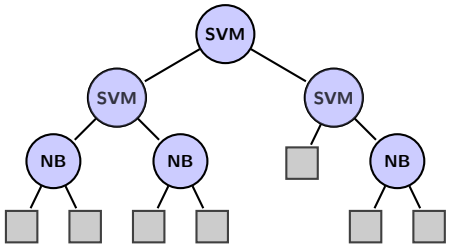


Figure 29: Hybird Classifier deployment using Adaptive Selection

Furthermore, depending on the relative priority to satisfy the conflicting constraints of accuracy and run-time, we can get best of both models by combining SVM and NB classifiers in an adaptive way. For node v in the hierarchy, this can be achieved by using a threshold τ_v for the feature set size to sample size ratio r_v. The threshold value τ_v determines the choice of the classifier in the following way

$$\text{Classifier at node } v = \begin{cases} \text{Naive Bayes} & \text{if } r_v \geq \tau_v \\ \text{SVM} & \text{otherwise} \end{cases}$$

The parameter $\tau = \{\tau_v\}, \forall v \in \mathcal{V}$, thus controls the tradeoff between accuracy of the overall classification system and the response time for training and prediction. Even though the above thresholding strategy is a simplification of the classifier selection criterion in section 4.3.1, it works well in practice as shown in our experiments and presented in more detail in section 4.3.3.

4.3.2 *Experimental Setup*

The experiments were performed on a Linux system with 24GB physical memory and 1TB hard-disk. We use the publicly available DMOZ data set from the LSHTC, 2011. The dataset, after having been preprocessed by stemming and stopword removal, appears in the LibSVM format. Table 9 presents the numeric values corresponding to the important properties of the dataset. Since the average number of labels per document is 1.02, we consider it as single-label classification problem for our purpose.

We use Liblinear Fan et al. [2008] to train the models for L2-regularized L2-loss support vector classification. The models are trained for all 7,574 non-leaf nodes in the hierarchy for One-Vs-All classification. For NB classifier, we implement the standard multinomial Naive Bayes using Laplace smoothing. Predictions are done in a top-down manner starting at the root node till the class corresponding to a leaf node is finally predicted.

Table 10 shows the different classification mechanisms to build the overall classifier, which include, (i) SVM classifier for the entire hierarchy, (ii) Adaptive

Property Name	Value
Total number of training examples	394,756
Size of the Overall Feature Space	594,158
Number of Target Classes ($\|\mathcal{Y}\|$)	27,875
Number of Nodes in the Hierarchy ($\|\mathcal{V}\|$)	35,449
Size of training file on Disk	586.3 MB
Depth of Hierarchy Tree	6
Total number of multiclass classifiers	7,574
Number of classifiers at depth 5	5,055

Table 9: Training Data Properties

Model employed	Accuracy in %	Training Time (hours)	Test Time (secs)
SVM for entire hierarchy	35.6	35	20
Adaptive Selection, $\tau = 60$	35.2	22	12
Adaptive Selection, $\tau = 30$	34.7	12	5
SVM with NB for last level	32.4	14	4
NB for entire hierarchy	22.2	0.25	0.5

Table 10: Tradeoff between Prediction Accuracy in %, Total Training for entire dataset in hours, and Average Test Time per Instance in seconds

classifier selection strategy based on threshold value, (iii) Static classifier selection by deploying NB classifier at lower levels, and finally (iv) NB classifier for the entire hierarchy. By employing SVM-only classification system, the accuracy (35.6%) is comparable to the best participant (38.8%) in LSHTC for the DMOZ track. However, we would like to point out that the objective of our work does not coincide with the participants' in the LSHTC challenge since the major focus of the challenge is on accuracy related metrics. As a result, some of the participants do not necessarily utilize the hierarchy completely as in Madani and Huang [2010] or may employ some post-processing for higher accuracy. On the other hand, we take a more principled approach leading to a more robust and interpretable analysis which is also applicable to other large scale hierarchical classification problems involving more complex topologies such as directed acyclic graphs. Moreover, we aim to study the tradeoffs involving various constraints which could be used to *tune* the desired behavior for a large scale hierarchical classification system.

4.3.3 *Results and Analysis*

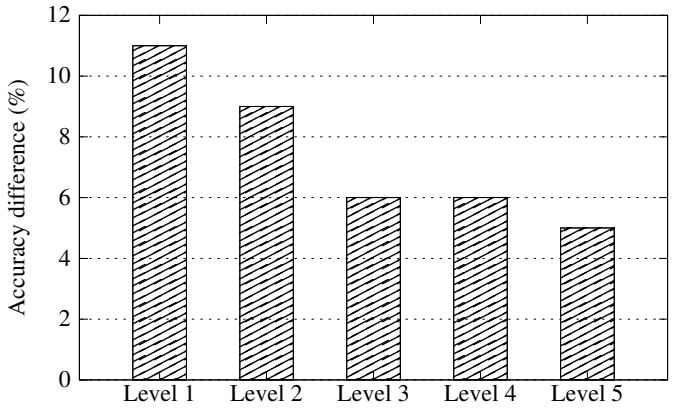


Figure 30: Difference of SVM and NB accuracy, (SVM - NB), in % for each hierarchy level. Level 1 corresponds to the root and level 5 to the level leading to leaves.

Table 10 shows the tradeoffs as we go from a fully discriminative framework to a fully generative one. When replacing the SVM classifiers (row 1) at the outer-most periphery of the hierarchy by NB (row 4), there is a 10% decrease in accuracy while the gain in prediction speed is close to 500%. This property could be leveraged to make robust real-time predictions such as for large scale Question-Answering systems or data stream environments which need real-time response for acceptable behavior. Also, there is an almost 3-fold improvement in training time as a result of this adaptation.

The gain in speed-up for training and test time is achieved as a result of more compact models built by NB as compared to SVM from same training data. All the NB models can, therefore, be loaded in the physical memory for predictions. For SVM, the total size of all the models is almost twice the physical memory size and hence the models for only the top two levels can be loaded in the physical memory.

The adaptive classifier selection as shown in row 2 and 3 of Table 10 was computed based on a uniform threshold value of $\tau_v = 60$ and $\tau_v = 30$, $\forall v \in \mathcal{V}$. Increasing the threshold value would select more SVM classifiers and thereby leading to better accuracy but slower training and test time. Decreasing it would correspond to more NB classifiers in the hierarchical framework, which leads to better run-time performance but lower accuracy.

Comparison between the adaptive classifier selection strategy and the static rule of applying NB classifier for the last level, rows 3 and 4 of Table 10, reveals another interesting observation. The prediction accuracy is noticeably higher

by employing the adaptive strategy, for comparable values of training and prediction time.

Figure 30 shows the variation of difference in accuracy of SVM and NB classifiers when plotted against levels in the hierarchy. As per the arguments given in section 4.3.1, SVM outperforms NB at the levels near the root node of the hierarchy. However, NB catches up with SVM for the classifiers at level 4 and level 5 of the hierarchy but it is not able to surpass SVM accuracy. This could be due to argument (1), i.e. $\varepsilon(f_{D,\infty}) \leq \varepsilon(f_{G,\infty})$, which implies that asymptotic generalization performance of SVM is better than that of NB.

4.3.4 *Remarks*

In this section, we proposed a method to combine SVM and NB classifier in a top-down cascade to address together the requirements of high prediction accuracy as well as prediction and training time. The proposed method is based on well founded theoretical results on the sample complexity of generative and discriminative classifiers. It also provides a parameter which can be used to tune the extent of the desired trade-off between prediction accuracy and computational complexity of training and prediction.

4.4 CONCLUSION

In this chapter, we presented applications of exploiting data distribution in large-scale web-taxonomies for designing machine learning algorithms. We focused on classification accuracy and training time in power-law distributed datasets consisting of rare categories. Our soft-thresholding based method aims to achieve higher values for the bound developed earlier in the chapter. The proposed method leads to improvement in classification accuracy and rare category detection for large-scale power-law distributed datasets. It not only performs better than state-of-art methods but is also easier to implement and efficient in terms of computational complexity. For large-scale datasets such as Wikipedia and Directory Mozilla, we use the developed bound further and propose an efficient alternative to k-fold cross-validation in these scenarios. This work can be seen as an instance of general paradigm of extracting latent information in Big Data to tackle the bottle-necks such as computational complexity of learning. Lastly, we presented tradeoffs between conflicting constraints of prediction accuracy and computing resources which are crucial for the design of large scale hierarchical classification systems. Our analysis was based on utilizing the heterogeneity in large scale web directories, such as DMOZ, for

designing effective local classifiers. We also presented an adaptive classifier selection strategy which can be employed to tune the extent of tradeoff.

5

FLAT VERSUS HIEARCHICAL CLASSIFICATION IN LARGE-SCALE TAXONOMIES

In this chapter, 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 by pruning the given taxonomy which 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.

5.1 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)[1], Wikipedia and Yahoo! Directory[2] are instances of such large scale textual datasets which consist of millions of documents which are distributed among hundreds of thousand target categories. Directory

1 www.dmoz.org

2 www.dir.yahoo.com

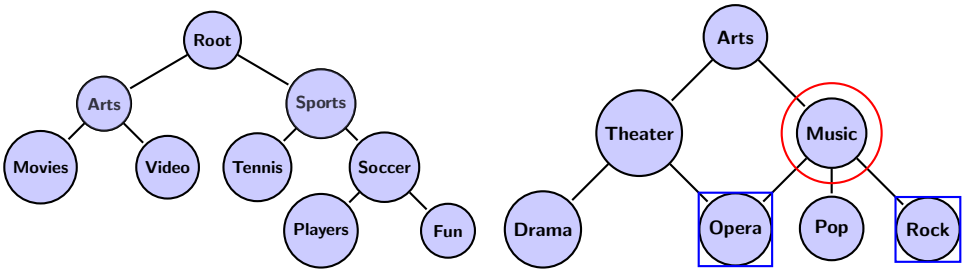


Figure 31: DMOZ and Wikipedia Taxonomies

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) [3] 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 is found in the taxonomies such as Wikipedia. The tree and DAG relationship among categories is illustrated for DMOZ and Wikipedia taxonomies in Figure 31.

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 which 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
- Designing the loss-function by taking hierarchy information into account
- Simplifying the given hierarchy, such by partially flattening the hierarchy

3 https://www.nlm.nih.gov/mesh/

- Ignoring the hierarchy information altogether and training flat classifiers, one for each target class

Hierarchical models for large scale classification however suffer from the fact that they have to make many decisions prior to reach 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 are likely to be confused while classifying a new document. Moreover, since the classification is not recoverable, it leads to the phenomena 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, potentially unbalanced. It is thus very difficult to assess which strategy is best and there is no consensus, at the time being, on to which approach, flat or hierarchical, should be preferred on a particular category system.

In this chapter, we study to address the problem of choosing between the two strategies from a learning-theoretic viewpoint. 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 and provide an explanation to several empirical findings reported in previous studies. Motivated by these bounds, 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. We also present approximation error based bounds for Logistic Regression and Naive Bayes classifiers deployed in large-scale taxonomies. Based on these bounds, we then propose a meta-learning strategy for hierarchy pruning which is applicable for both discriminative and generative classifiers. With the aim of synchronizing the taxonomy with the training set comprising of the set of input-output pairs, we provide a detailed analysis of classification accuracy for both the hierarchy pruning strategies. Contrary to Dekel [2009] that reweighs the edges in a taxonomy through a cost sensitive loss function to achieve this goal, we use here a simple pruning strategy that modifies the taxonomy in an explicit way.

Th chapter is organized as follows: In section 5.2 we review the recently proposed approaches in the context of large-scale hierarchical text classification. Since the formal framework presented in this chapter is based on Rademacher complexity, we recall the concepts related to function class complexity in Section 5.3. We refer to the excellent text by Mohri et al. [2012] in order to present the background related concepts. We introduce the notations used in Section 5.4 and then study flat versus hierarchical strategies by studying

the generalization error bounds for classification in large-scale taxonomies. Approximation error for multi-class versions of Naive Bayes and Logistic Regression classifiers are presented in Section 5.5.1 and Section 5.5.2 respectively. Based on these bounds, the two pruning strategies are presented in Section 5.4.2 and Section 5.5.3 Section 5.6 illustrates these developments via experiments conducted on several taxonomies extracted from DMOZ and the International Patent Classification. The experimental results are in line with results reported in previous studies, as well as with our theoretical developments. Finally, Section 5.7 concludes this study.

5.2 RELATED WORK

Large-scale classification, involving tens of thousand target categories, has assumed significance importance 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) and Large Scale Visual Recognition Challenge (ILSVRC) 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 in 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 in 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. Deep Classification 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 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 in 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 structure. However, these approaches were applied to the datasets in which the number of categories were limited to a few hundreds. Recent approaches wherein target categories in the range of thousands and beyond have been proposed which include Bayesian modeling of large scale hierarchical classification 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. Also, recursive-regularization based strategy for large-scale classification has been proposed in Gopal and Yang [2013b].

Hierarchy simplification by flattening entire layer in the hierarchy has been studied from an empirical view-point in Wang and Lu [2010], Malik [2009]. These strategies for taxonomy adaptation by flattening 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 for taxonomy adaptation has the advantage that, (i) it is based on a well-founded theoretical criteria, and (ii) its application in a node-specific sense rather than applying to an entire layer. The study in 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.

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 such as Bengio et al. [2010], Beygelzimer et al. [2009], Gao and Koller [2011]. 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 in 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 while not sacrificing the training and prediction speed.

5.3 RADEMACHER COMPLEXITY : A REVIEW

In this section, we review some concepts related to Rademacher complexity which is a framework to measure the complexity of a function class.

Let $\mathcal{X}$ denote an input space and $\mathcal{Y}$ denote the set of target labels. Let function class $\mathcal{F}$ and $f \in \mathcal{F}$ maps an input $\mathbf{x} \in \mathcal{X}$ to $\mathcal{Y}$. Also, by $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ we denote a loss function. For each $f \in \mathcal{F}$, we can associate a function g that maps $(\mathbf{x}, y) \in (\mathcal{X} \times \mathcal{Y})$ to $L(f(\mathbf{x}), y)$. We denote by G *the family of loss functions associated to* the function class $\mathcal{F}$.

In the light of the above setup, the Rademacher of a function class can be seen as its ability to fit random noise. Higher the Rademacher complexity of a function class, more likely it is to overfit. Formally, it is defined as follows:

Definition 1. ***Empirical Rademacher Complexity*** *Mohri et al. [2012]*
Let G be a family of functions mapping from Z to $[a, b]$ and $S = (z_1, \ldots, z_m)$ a fixed sample of size m with elements in Z. Then, the empirical Rademacher complexity of G with respect to the sample S is given by

$$\hat{\mathcal{R}}_m(G) = \mathbb{E}_\sigma \left[\sup_{g \in G} \frac{1}{m} \sum_{i=1}^{m} \sigma_i g(z_i) \right] \tag{5.3.1}$$

where $\sigma = (\sigma_1, \ldots, \sigma_m)^T$, where σ_i are independent random variables each having value in $\{-1, +1\}$ with equal probability.

For the finite sample S, let g_S denote the vector of values taken by the function g, i.e., $g_S = (g(z_1), \ldots, g(z_m))^T$. Then, Rademacher complexity of the function class G is also denoted by

$$\hat{\mathcal{R}}_m(G) = \mathbb{E}_\sigma \left[\sup_{g \in G} \frac{\langle \sigma, g_S \rangle}{m} \right] \tag{5.3.2}$$

where $\langle , \rangle$ denotes the dot product and is a measure of correlation between g_S and random noise. The supremum denoted by $\sup_{g \in G} \frac{\langle \sigma, g_S \rangle}{m}$ is a measure of degree of correlation of random noise with the function class G.

Definition 2. ***Rademacher Complexity*** *Mohri et al. [2012]*
For a sample size $m \geq 1$, the Rademacher complexity of G, denoted $\mathcal{R}_m(G)$, is the expectation of the empirical Rademacher complexity over all samples of size m which are drawn from the underlying distribution $\mathcal{D}$, i.e.,

$$\mathcal{R}_m(G) = \mathbb{E}_{S \sim \mathcal{D}^m} \left[\hat{\mathcal{R}}_m(G) \right] \tag{5.3.3}$$

Based on the above notions of Rademacher complexity of a function class, we now give a standard result which relates the expected value of the composing functions to the empirically observed value and the Rademacher complexity of the function class.

Theorem 1. *Mohri et al. [2012] Let G be a family of functions mapping from Z to $[0,1]$. Then, for any $\delta > 0$, with probability atleast $1-\delta$, each of the following holds $\forall g \in G$:*

$$\mathbb{E}\left[g(z)\right] \leq \frac{1}{m}\sum_{i=1}^{m} g(z_i) + 2\mathcal{R}_m(G) + \sqrt{\frac{\log\frac{1}{\delta}}{2m}} \tag{5.3.4}$$

$$\mathbb{E}\left[g(z)\right] \leq \frac{1}{m}\sum_{i=1}^{m} g(z_i) + 2\hat{\mathcal{R}}_m(G) + 3\sqrt{\frac{\log\frac{2}{\delta}}{2m}} \tag{5.3.5}$$

Based on the above results for Rademacher complexity of a function class, we now present a generalization error bound for classifier deployed in a taxonomy in a top-down manner. We then compare it to the standard result for generalization error bound for flat multi-class classification and attempt to address the problem of flat versus hierarchical classification in large-scale classification.

5.4 FLAT VS HIERARCHICAL CLASSIFICATION : A LEARNING THEORETIC VIEW-POINT

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 $\pi : V \setminus \{\perp\} \rightarrow V$ 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 sisters $\mathfrak{S}(v) = \{v' \in V \setminus \{\perp\}; v \neq v' \wedge \pi(v) = \pi(v')\}$ and its daughters $\mathfrak{D}(v) = \{v' \in V \setminus \{\perp\}; \pi(v') = v\}$. 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 multiclass classification problem.

5.4.1 *A hierarchical Rademacher data-dependent bound*

Our main result is the following theorem which provides a data-dependent bound on the generalization error of a top-down multiclass hierarchical classifier. We consider here kernel-based hypotheses, with $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ a PDS kernel and $\Phi : \mathcal{X} \to \mathbb{H}$ 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].

Theorem 2. *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 $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a PDS kernel and let $\Phi : \mathcal{X} \to \mathbb{H}$ 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 multiclass classification generalization bound holds for all $g_f \in \mathcal{G}_{\mathcal{F}_B}$:*

$$\mathcal{E}(g_f) \leq \frac{1}{m} \sum_{i=1}^{m} \mathcal{A}(g_f(\mathbf{x}^{(i)}, y^{(i)})) + \frac{8BRL}{\sqrt{m}} \sum_{v \in V \setminus \mathcal{Y}} |\mathfrak{D}(v)|(|\mathfrak{D}(v)| - 1) + 3\sqrt{\frac{\ln(2/\delta)}{2m}} \tag{5.4.1}$$

where $|\mathfrak{D}(v)|$ denotes the number of daughters 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:

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

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:

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

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 in Guermeur [2010] for flat multiclass 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:

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

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:

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

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

For flat multiclass classification, we recover the bounds of Guermeur [2010] by considering a hierarchy containing a root node with as many daughters 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 in Guermeur [2010], and that the factor $8L$ in the complexity term of the bound, instead of 4 in 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.

Flat vs hierarchical classification in large-scale taxonomies. The generalization error is controlled in inequality (5.4.1) 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 2 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 2 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 5.6.

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 (5.4.1): 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 2, 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.

5.4.2 *Lowering the bound by hierarchy pruning*

In this section, we present a strategy which aims to adapt the given hierarchy of classes by pruning some nodes in the hierarchy. An example of node pruning is shown in Figure (figure 32). The rationale behind adapting the given hierarchy $\mathcal{H} = (V, E)$ to the set of input/output pair $(\mathbf{x}, y)$ is that

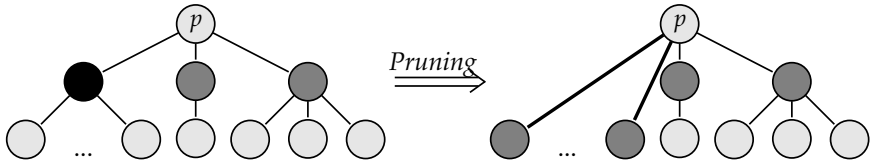


Figure 32: The pruning procedure; the node in black is replaced by its children.

- 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
- The large-scale nature of such taxonomies poses difficulties in manually designing good taxonomies for classification.

In view of the generalization error bound derived in Theorem 2, adapting the given taxonomy of classes is aimed 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 derived in Theorem 2 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 2, the empirical Rademacher complexity was upper bounded as follows:

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

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

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

Essentially, $C(v)$ denotes the confusion of node v with its sibling nodes. This is so, since *more* the category denoted by node v is confused with its siblings,

lower the attained margin by the separating hyper-plane and hence, *higher* the norm given by $\|\mathbf{w}_v\|_{\mathbb{H}}$. The above bound suggests that the error due to Rademacher complexity term can be reduced by pruning those nodes v in the taxonomy for which $C(v)$ is maximal. This strategy identifies the candidate nodes which when pruned lead to decrease in the error due to propagation at cost of minimum increase in the error due to Rademacher complexity. Pruning the most confused nodes leads to *short-circuiting* those root-to-leaf paths which are likely to lead to classification error. In practice, we focus on pruning the nodes in the top-two layers of the taxonomy since nodes in these levels represent generic categories which are typically over-lapping in nature. The pruning process as an algorithmic procedure is shown in Algorithm 4, where the variable Δ is used to stop the pruning process in an inner iteration.

Algorithm 4 The proposed method for hierarchy pruning

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
    $\Delta = C(v'_1) - C(v'_1)$
    $v_{prev} \leftarrow v'_2$ ▷ 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 above criterion for pruning the nodes in a large-scale taxonomy is also similar in spirit to the method introduced in Babbar et al. [2013b] which is motivated from the generalization error analysis of Perceptron Decision Trees Bennett et al. [2000]. Furthermore, this is also related to margin-based techniques for *construction of taxonomies* as developed in Bengio et al. [2010], Gao and Koller [2011], Yang and Tsang [2011]. 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. However, this method of hierarchy pruning 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 also is applicable to both discriminative and generative classifiers.

5.5 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 is combined in a typical top-down cascade. 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 32) will improve the classification accuracy or not.

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 3. *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*

$$\mathcal{E}(h) \leq \mathcal{E}(h_\infty) + G\left(O\left(\sqrt{\frac{1}{m}\log d}\right)\right) \tag{5.5.1}$$

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 in Ng and Jordan [2001].

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

$$l(\boldsymbol{x}, y_k) = \log\left[\widehat{P}(y_k)\prod_{i=1}^d \widehat{P}(w_i|y_k)^{x_i}\right] \tag{5.5.2}$$

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:

$$h(\boldsymbol{x}) = \underset{y_k \in \mathcal{Y}}{\operatorname{argmax}}\, l(\boldsymbol{x}, y_k) \tag{5.5.3}$$

and the one for its asymptotic version by:

$$h_\infty(\boldsymbol{x}) = \underset{y_k \in \mathcal{Y}}{\operatorname{argmax}}\, l_\infty(\boldsymbol{x}, y_k) \tag{5.5.4}$$

Lemma 2 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 4 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 3. *Let $l_\infty^1(\boldsymbol{x}) = \max_{y_k \in \mathcal{Y}} l_\infty(\boldsymbol{x}, y_k)$ be the best log-likelihood score obtained for $\boldsymbol{x}$ by the asymptotic Naive Bayes classifier, and let $l_\infty^2(\boldsymbol{x}) = \max_{y_k \in \mathcal{Y} \setminus h_\infty(\boldsymbol{x})} l_\infty(\boldsymbol{x}, y_k)$ 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:*

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

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 4. *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 $(1 - \delta_{\mathcal{Y}})$:*

$$\mathcal{E}(h) \leq \mathcal{E}(h_\infty) + G_{\mathcal{Y}}(\epsilon) \tag{5.5.5}$$

with:

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

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 (sketch) Using Lemma 2 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 2, $\sqrt{C}$ 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 ($\epsilon' = \epsilon\sqrt{C}(n + d_{max})$) and approximating $\sum_{k=1}^{K} \exp(-2n_k\epsilon^2)$ by $\exp(-2d_{min}\epsilon^2)$, the dominating term in the sum, leads to the desired result. □

5.5.2 *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 :

$$h_m(\mathbf{x}) = \operatorname*{argmax}_{y\in\mathcal{Y}} P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_m) \tag{5.5.6}$$

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 5.4 wherein these represented Rademacher random variables.

Proof (sketch) 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 5 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 :

$$h_\infty(\mathbf{x}) = \operatorname*{argmax}_{y \in \mathcal{Y}} P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_\infty) \tag{5.5.7}$$

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

Definition 4. *Let $f^1_\infty(\mathbf{x}) = \max_{y \in \mathcal{Y}} P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_\infty)$ be the best class posterior probability for $\mathbf{x}$ by the asymptotic MLR classifier, and let $f^2_\infty(\mathbf{x}) = \max_{y \in \mathcal{Y} \setminus h_\infty(\mathbf{x})} P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_\infty)$ 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:*

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

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 5. *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:*

$$\mathcal{E}(h_m) \leq \mathcal{E}(h_\infty) + G_{\mathcal{Y}}\left(d\sqrt{\frac{R|\mathcal{Y}|\sigma_0}{\delta m}}\right) \tag{5.5.8}$$

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 (sketch) 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 5 represents the largest value of the inverse (diagonal) Fisher information matrix (Schervish [1995]). It is thus the

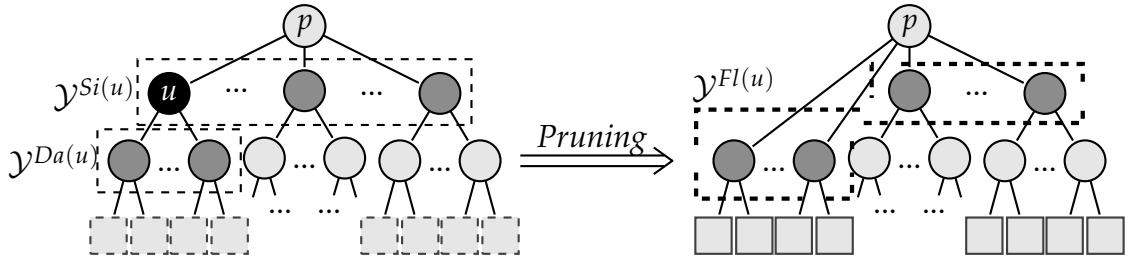


Figure 33: The pruning procedure for a candidate class node u (in black). After replacing the candidate node by its children, the new category set $\mathcal{Y}^{Fl(u)}$ contains the classes from both the daughter and the sister category sets of u.

smallest value of the (diagonal) Fisher information matrix, and is related 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.

5.5.3 *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 sister categories with the node itself $\mathfrak{S}'(v) = \mathfrak{S}(v) \cup \{v\}$, its daughter categories, $\mathfrak{D}(v)$, and the union of its sister and daughter 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 the `MLR` classifier by $h_m^{\mathfrak{S}'_v}$ learned from a set of sister categories of node v and the node itself, and by $h_m^{\mathfrak{D}_v}$ a `MLR` classifier learned from the set of daughter 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 Theorem 5 to this setting.

Theorem 6. *With the notations defined above, for* `MLR` *classifiers,* $\forall \epsilon > 0$, $v \in V \setminus \mathcal{Y}$, *one has, with probability at least* $1 - \left(\frac{Rd^2 |\mathfrak{S}'(v)| \sigma_0^{\mathfrak{S}'(v)}}{m_{\mathfrak{S}'(v)} \epsilon^2} + \frac{Rd^2 |\mathfrak{D}(v)| \sigma_0^{\mathfrak{D}(v)}}{m_{\mathfrak{D}(v)} \epsilon^2} \right)$:

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

$\{|\mathcal{Y}^\ell|, m_{\mathcal{Y}^\ell}, \sigma_0^{\mathcal{Y}^\ell}; \mathcal{Y}^\ell \in \{\mathfrak{S}'(v), \mathfrak{D}(v)\}\}$ 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 5. Denoting by $h_m^{\mathfrak{F}v}$ the MLR classifier trained on the set $\mathfrak{F}(v)$ and by $h_\infty^{\mathfrak{F}v}$ its asymptotic version, Theorem 6 suggests that one should prune node v if:

$$G_{\mathfrak{F}(v)}(\epsilon) \leq G_{\mathfrak{S}'(v)}(\epsilon) + G_{\mathfrak{D}(v)}(\epsilon) \text{ and } \frac{|\mathfrak{F}(v)|\sigma_0^{\mathfrak{F}(v)}}{m_{\mathfrak{F}(v)}} \leq \frac{|\mathfrak{S}'(v)|\sigma_0^{\mathfrak{S}'(v)}}{m_{\mathfrak{S}'(v)}} + \frac{|\mathfrak{D}(v)|\sigma_0^{\mathfrak{D}(v)}}{m_{\mathfrak{D}(v)}} \tag{5.5.9}$$

Furthermore, the bounds obtained 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 in the logistic regression family is close or not to its asymptotic version. Each node $v \in V$ can then be characterized by factors in the set $\{|\mathcal{Y}^\ell|, m_{\mathcal{Y}^\ell}, d_{max}^{\mathcal{Y}^\ell}, d_{min}^{\mathcal{Y}^\ell}, G_{\mathcal{Y}^\ell}(.)|\mathcal{Y}^\ell \in \{\mathfrak{S}'(v), \mathfrak{D}(v), \mathfrak{F}(v)\}\}$ which are involved in the estimation of inequalities (5.5.9) above. We propose to estimate the confusion term $G_{\mathcal{Y}^\ell}(.)$ with two simple quantities: the average cosine similarity of all the pairs of classes in $\mathcal{Y}^\ell$, and the average symmetric Kullback-Leibler divergences between all the pairs in $\mathcal{Y}^\ell$ of class conditional multinomial distributions.

Algorithm 5 presents the process of learning the hierarchy pruning by learning a meta-classifiers 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 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 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.

5.6 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

Algorithm 5 The pruning strategy.

```
procedure PRUNE HIERARCHY(a hierarchy H, a meta-classifier C_m)
    clist[] ← H.root;                                  ▷ Initialize with root node
    for j = 1 ... clist.size() do
        list[] ← Ch(clist[j]);                         ▷ Candidate children
        while !list.isEmpty() do
            index ← MERGE(clist[j],list,C_m);
            if index == null then
                break;
            end if
            list.remove(index);
        end while
        clist.add(Ch(clist[j]));                       ▷ Adds next level parents
    end for
    export new hierarchy;
end procedure
function MERGE(a parent p, list of children L,C_m)
    max ← −Double.MAX
    for i = 1 ... L.size() do
        ins ← createMetaInstance(p, L[i]);
        probs[] ← C_m(ins);
        if probs[0] > max then
            max ← probs[0]  ▷ The prob. for the positive class is stored in probs[0]
            index ← i;
        end if
        if max > 0.5 then
            merge p and L[index];
            return index;
        end if
    end for
    return null;
end function
```

Patent Classification (**IPC**) dataset[4] and the publicly available DMOZ dataset from the second PASCAL large scale hierarchical text classification challenge (**LSHTC2**)[5]. Both datasets are multi-class; **IPC** is single-label and **LSHTC2** multi-label with an average of 1.02 categories per class. We created 4 datasets from **LSHTC2** by splitting randomly the first layer nodes (11 in total) of the

4 http://www.wipo.int/classifications/ipc/en/support/

5 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 11: 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.

original hierarchy in disjoint subsets. The classes for the **IPC** and **LSHTC2** datasets are organized in a hierarchy in which the documents are assigned to the leaf categories only. Table 11 presents the characteristics of the datasets.

CR denotes the complexity ratio between hierarchical and flat classification, given by the Rademacher complexity term in Theorem 2: $\left(\sum_{v \in V \setminus \mathcal{Y}} |\mathfrak{D}(v)|(|\mathfrak{D}(v)| - 1)\right) / (|\mathcal{Y}|(|\mathcal{Y}| - 1))$; 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 hierarchal 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 11) 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

	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	**44.6**
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$^{\downarrow}$	45.8$^{\downarrow\downarrow}$
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$^{\downarrow}$	46.5$^{\downarrow\downarrow}$
PR-B	-	**48.1**	**49.5**	-	**46.6**	**46.5**	-	**52.2**	**52.2**	-	**54.5**	45.0
PR-M	**61.3**	**48.0**	**49.3**	**65.4**	**46.9**	**47.2**	**64.8**	**52.2**	**52.3**	**63.9**	**54.4**	45.0

Table 12: 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.

$\{+1; -1\}$. 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 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 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.

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 5.4.2 and (ii) Meta-learning based pruning strategy (PR-M) proposed in Algorithm 5. For the random pruning we restrict the procedure to the first two levels and perform 4 random prunings (this is the average number of prunings that are performed in the PR-M and PR-B strategies). 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 apply the $L2$-regularized versions, setting the penalty parameter C by cross-validation.

	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	**45.6**
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$^{\downarrow\downarrow}$	43.8$^{\downarrow\downarrow}$
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$^{\downarrow}$	42.4$^{\downarrow}$
PR-B	-	**33.1**	**32.3**	-	**34.7**	**34.4**	-	**31.8**	**31.9**	-	**48.1**	43.8
PR-M	**22.4**	**33.2**	**32.4**	21.2	**34.8**	**34.3**	**19.3**	**31.7**	**31.8**	**26.5**	**48.2**	43.7

Table 13: 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.

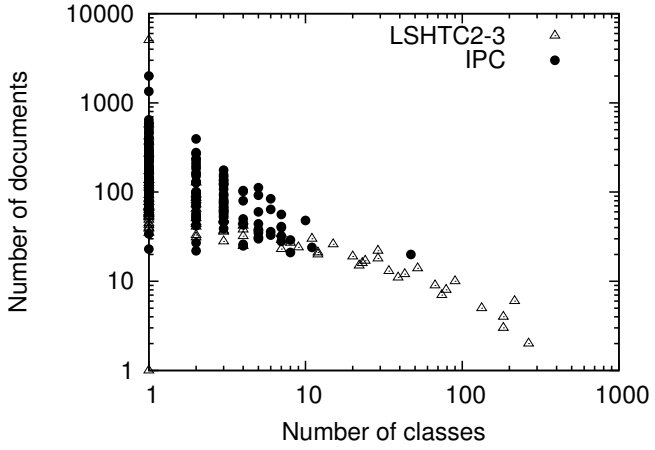


Figure 34: Number of classes (on X-axis) which have the specified number of documents (on Y-axis) for **LSHTC2-3** dataset and **IPC** dataset

5.6.1 *Flat versus Hierarchical classification*

The accuracy results (Micro-F1 measure) on **LSHTC2-3,4,5** and **IPC** are reported in Table 12. 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 11 as well as with the results obtained in Liu et al. [2005], Dumais and Chen [2000] over the same type 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.

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 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 34 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. 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, as Bengio et al. [2010], Gao and Koller [2011], Perronnin et al. [2012], Deng et al. [2011], in which the datasets considered do not have *rare categories* and are more well-balanced. The class-based performance (Macro-F1 measure) are given in Table 13.

5.6.2 *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 two strategies, one based on minimizing the rademacher-based generalization error bound (PR-B) and another based on meta-learning (PR-M) against the random pruning (RN) and fully hierarchical (FH) classification. As shown in Table 12, 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. 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.

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

(PR-M). 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.

5.6.3 *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 MNB and MLR are depicted in Figures 35 and 36, 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 prunings 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 where the nodes are pruned and their sequence. For example, in dataset **LSHTC2-4** (Figure 35), 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 36 and across

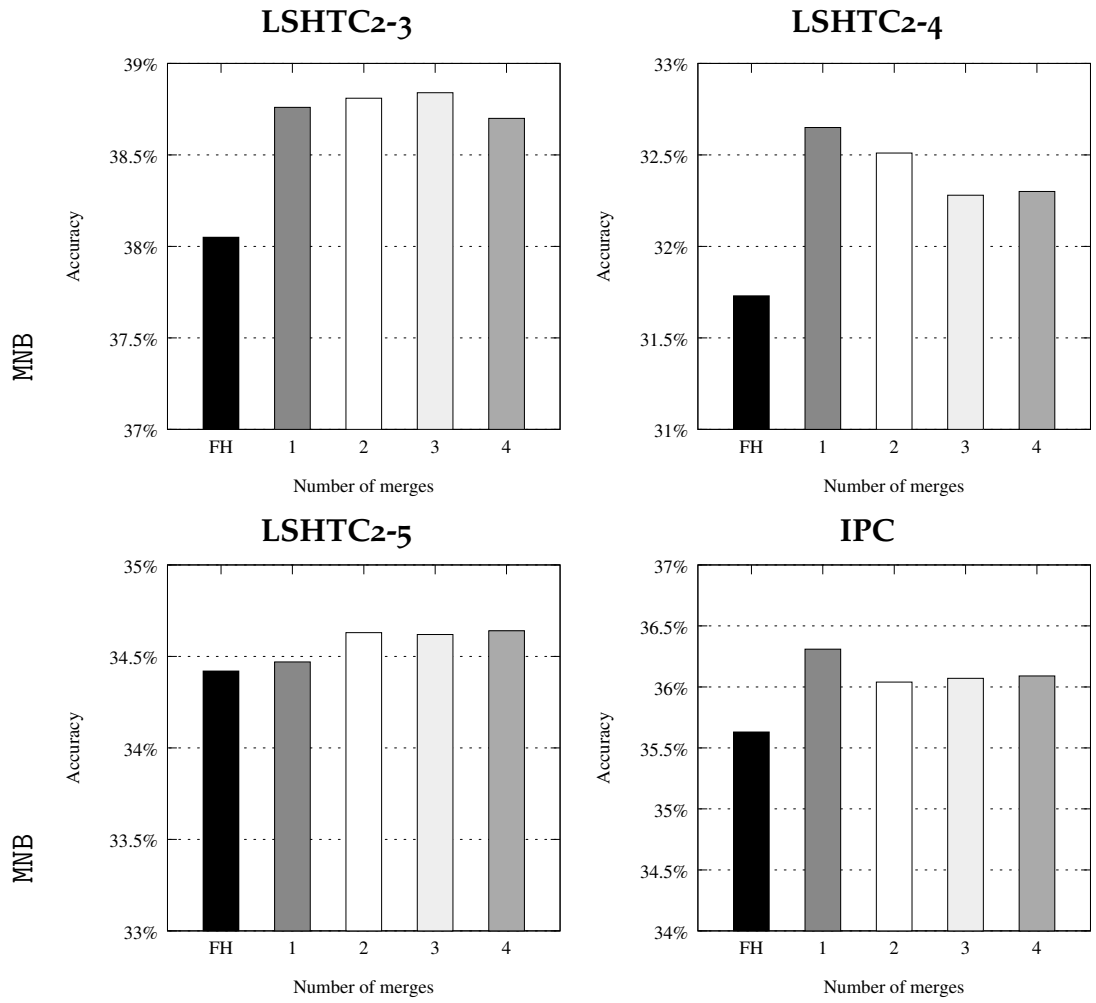


Figure 35: Accuracy performance with respect to the number of pruned nodes for MNB on different test sets.

all datasets that the behavior of the pruning method is more stable without decrease in the final performance.

5.7 CONCLUSION

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

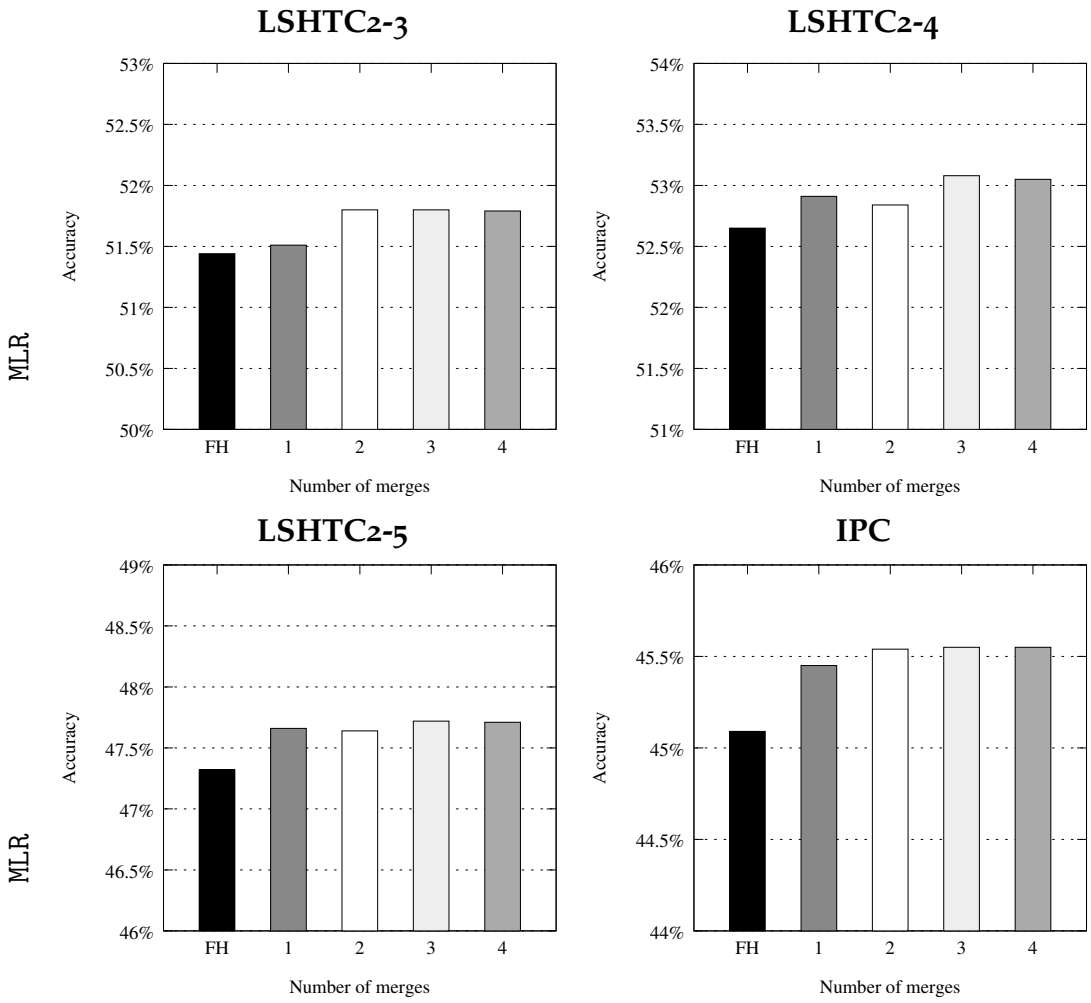


Figure 36: Accuracy performance with respect to the number of pruned nodes for MLR (down) on different test sets.

meta-learning technique which is based on the features derived from from the approximation-error based generalization bounds proposed in Sections 5.5.1 and 5.5.2. The experimental results reported here (as well as in the previous works) 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 (*e.g.* Bengio et al. [2010]). We plan to explore this in future work.

6

CONCLUSION AND PERSPECTIVES

In the era of Big Data, we need efficient and scalable machine learning algorithms which can perform automatic classification of Tera-Bytes of data in large-scale category systems. In Chapter 1, we discussed such category systems including Yahoo! directory, Wikipedia, Amazon Product Hierarchy and National Library of Medicine among others. Therein, we also presented some of the research challenges associated with large-scale supervised classification. In addition to the computational complexity of training and prediction, the test set performance of state-of-the-art classification algorithms suffers due to the power-law distribution in most naturally occurring large-scale datasets. Furthermore, being able to detect rare categories remains a practical challenge for such datasets. We covered some of the important state-of-the-art methods to address these problems in Chapter 2.

In Chapter 3, we studied the generative mechanisms in large-scale taxonomies which lead to power-law distribution of documents among categories. This was based on the famous Yule's model and model based on Preferential attachment. This study offers useful insights about the structure of large-scale web-directories. Furthermore, we used the fit to power-law distribution to study the space complexity of large-scale hierarchical classification systems.

We further leverage the distribution of data in large-scale category systems, and in Chapter 4, we have presented algorithms to tackle some of the challenges in large-scale learning. The soft-thresholding based classification method not only leads to better performance when measured by Micro-F1 and Macro-F1 measures but achieves this at a much lower computational cost as compared to the state-of-the-art methods. We also proposed an efficient model selection method for determining the regularization parameter in learning One-vs-Rest SVM classifier for large-scale power-law distributed category systems.

Finally, we address another key model selection problem in large scale classification concerning the choice between flat versus hierarchical classification from a learning theoretic aspect. The presented generalization error analysis provides an explanation to empirical findings in many recent studies in large-scale hierarchical classification. We further exploit the developed bounds to propose two methods for adapting the given taxonomy of categories to output taxonomies which yield better test accuracy when used in a top-down setup.

Large-scale learning is a relatively recent phenomena in the field of machine learning and offers interesting research directions. From the point of the work presented in this thesis, there are certainly some perspectives for future work. Building a taxonomy of categories from ground-up has been proposed in the form of computationally-intensive approaches such as Bengio et al. [2010], Gao and Koller [2011], Deng et al. [2011]. In this direction, our generalization error analysis can possibly be extended to design efficient mechanisms for building hierarchies. The trade-off between the empirical error and rademacher complexity for a top-down classifier can be used to group similar categories together while restricting the depth of the tree at the same time. Furthermore, PAC-Bayesian analysis could be applied to study the model selection problem of selecting the regularization parameter in large-scale linear SVM. This would eliminate the need for test set while selecting the regularization parameter.

Also, we have focused on single-labels datasets in this thesis, one interesting extension of this work is to address the problems in multi-labeled domains. In this respect, extending the theoretical framework in Chapter 5, to multi-label classification setting can be challenging since there can be more than one correct root-to-leaf paths. Effective detection of *rare categories* in large-scale learning remains an important challenge. In this direction, formalization of soft-thresholding based framework by using the power-law distribution as prior knowledge can lead to interesting solutions which are specialized for detecting rare categories.

REFERENCES

D.J. Newman A. Asuncion. UCI machine learning repository, 2007. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

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.

Erin L Allwein, Robert E Schapire, and Yoram Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *The Journal of Machine Learning Research*, 1:113–141, 2001.

Rohit Babbar, Ioannis Partalas, Eric Gaussier, and Cecile Amblard. On empirical tradeoffs in large scale hierarchical classification. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 2299–2302. ACM, 2012.

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 Berlin Heidelberg, 2013b.

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, 2014.

Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *science*, 286(5439):509–512, 1999.

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, Michael I Jordan, and Jon D McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

Samy Bengio, Jason Weston, and David Grangier. Label embedding trees for large multi-class tasks. In *Neural Information Processing Systems*, 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 international ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 11–18, 2009.

Alina Beygelzimer, John Langford, and Pradeep Ravikumar. Multiclass classification with filter trees.

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, 2009. AUAI Press.

Wei Bi and James T Kwok. Multi-label classification on tree-and dag-structured hierarchies. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 17–24, 2011.

Wei Bi and James T Kwok. Hierarchical multilabel classification with minimum bayes risk. In *Proceedings of the 2012 IEEE 12th International Conference on Data Mining*, pages 101–110. IEEE Computer Society, 2012a.

Wei Bi and James T Kwok. Mandatory leaf node prediction in hierarchical multilabel classification. In *Advances in Neural Information Processing Systems*, pages 153–161, 2012b.

Christopher M Bishop et al. *Pattern recognition and machine learning*, volume 1. springer New York.

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

Léon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In *Advances In Neural Information Processing Systems*, pages 161–168, 2008.

Olivier Bousquet and André Elisseeff. Stability and generalization. *The Journal of Machine Learning Research*, 2:499–526, 2002.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2009.

Lijuan Cai and Thomas Hofmann. Hierarchical document categorization with support vector machines. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 78–87, 2004.

Andrea Capocci, Vito DP Servedio, Francesca Colaiori, Luciana S Buriol, Debora Donato, Stefano Leonardi, and Guido Caldarelli. Preferential attachment in the growth of social networks: The internet encyclopedia wikipedia. *Physical Review E*, 74(3):036116, 2006.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *arXiv preprint arXiv:1106.1813*, 2011.

Jianfu Chen and David Warren. Cost-sensitive learning for large-scale hierarchical classification. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 1351–1360. ACM, 2013.

Moustapha Cissé, Thierry Artières, and Patrick Gallinari. Learning compact class codes for fast inference in large multi class classification. In *Machine Learning and Knowledge Discovery in Databases*, pages 506–520. Springer, 2012.

Moustapha M Cisse, Nicolas Usunier, Thierry Artières, and Patrick Gallinari. Robust bloom filters for large multilabel classification tasks. In *Advances in Neural Information Processing Systems*, pages 1851–1859, 2013.

Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, pages 265–292, 2002.

John Shawe-Taylor Nello Cristianini. Data-dependent structural risk minimisation for perceptron decision trees. In *Advances in Neural Information Processing Systems 10: Proceedings of the 1997 Conference*, volume 10, page 336. MIT Press, 1998.

Amit Daniely, Sivan Sabato, and Shai S Shwartz. Multiclass learning approaches: A theoretical comparison with implications. In *Advances in Neural Information Processing Systems*, pages 485–493, 2012.

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.

Luc Devroye. *A probabilistic theory of pattern recognition*, volume 31. springer, 1996.

Thomas G Dietterich and Ghulum Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2 (263):286, 1995.

Sergey N Dorogovtsev and José Fernando F Mendes. Evolution of networks with aging of sites. *Physical Review E*, 62(2):1842, 2000.

Susan Dumais and Hao Chen. Hierarchical classification of web content. In *Proceedings of the* 23^{rd} *Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, pages 256–263, 2000.

Leo Egghe. Untangling herdan's law and heaps' law: Mathematical and informetric arguments. *Journal of the American Society for Information Science and Technology*, 58(5):702–709, 2007.

Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. *SIGCOMM*.

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.

Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.

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.

Markus M Geipel, Claudio J Tessone, and Frank Schweitzer. A complementary view on the growth of directory trees. *The European Physical Journal B*, 71(4): 641–648, 2009.

Siddharth Gopal and Yiming Yang. Distributed training of large-scale logistic models. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 289–297, 2013a.

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, 2013b.

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

Yann Guermeur. Vc theory of large margin multi-category classifiers. *The Journal of Machine Learning Research*, 8:2551–2594, 2007.

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

Maya R Gupta, Samy Bengio, and Jason Weston. Training highly multiclass classifiers. *Journal of Machine Learning Research*, 15:1–48, 2014.

Zaid Harchaoui, Matthijs Douze, Mattis Paulin, Miroslav Dudik, and Jérôme Malick. Large-scale image classification with trace-norm regularization. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3386–3393. IEEE, 2012.

Bharath Hariharan, Lihi Zelnik-Manor, Manik Varma, and SVN Vishwanathan. Large scale max-margin multi-label classification with priors. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 423–430, 2010.

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

Trevor Hastie, Saharon Rosset, Robert Tibshirani, and Ji Zhu. The entire regularization path for the support vector machine. In *Journal of Machine Learning Research*, pages 1391–1415, 2004.

Cho-Jui Hsieh, Kai-Wei Chang, Chih-Jen Lin, S Sathiya Keerthi, and Sellamanickam Sundararajan. A dual coordinate descent method for large-scale linear

svm. In *Proceedings of the 25th international conference on Machine learning*, pages 408–415. ACM, 2008.

Thorsten Joachims. Making large scale svm learning practical. 1999.

Qi Ju and Alessandro Moschitti. Incremental reranking for hierarchical text classification. In *Advances in Information Retrieval*, pages 726–729. Springer, 2013.

S Sathiya Keerthi, Sellamanickam Sundararajan, Kai-Wei Chang, Cho-Jui Hsieh, and Chih-Jen Lin. A sequential dual method for large scale multi-class linear svms. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 408–416. ACM, 2008.

Konstantin Klemm, Víctor M Eguíluz, and Maxi San Miguel. Scaling in the structure of directory trees in a computer cluster. *Physical review letters*, 95 (12):128701, 2005.

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.

Aris Kosmopoulos, Ioannis Partalas, Eric Gaussier, Georgios Paliouras, and Ion Androutsopoulos. Evaluation measures for hierarchical classification: a unified view and novel approaches. *arXiv preprint arXiv:1306.6802*, 2013.

Chih-Jen Lin, Ruby C Weng, and S Sathiya Keerthi. Trust region newton method for logistic regression. *The Journal of Machine Learning Research*, 9:627–650, 2008.

Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

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*, 2005.

O. Madani and J. Huang. Large-scale many-class prediction via flat techniques. *Workshop on Large-Scale Hierarchical Text Classification at ECIR*, 2010.

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

Benoit Mandelbrot. A note on a class of skew distribution functions: Analysis and critique of a paper by ha simon. *Information and Control*, 2(1):90–99, 1959.

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.

Cornelia Metzig and Mirta B Gordon. A model for scaling in firms' size and growth rate distribution. *Physica A*, 2014.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. The MIT Press, 2012.

Yurii Nesterov. Primal-dual subgradient methods for convex problems. *Mathematical programming*, 120(1):221–259, 2009.

M. E. J. Newman. Power laws, Pareto distributions and Zipf's law. *Contemporary Physics*, 2005a.

MEJ Newman. Power laws, pareto distributions and zipf's law. *Contemporary Physics*, 46(5):323–351, 2005b.

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.

Ioannis Partalas, Rohit Babbar, Eric Gaussier, and Cecile Amblard. Adaptive classifier selection in large-scale hierarchical classification. In *Neural Information Processing*, pages 612–619. Springer, 2012.

Florent Perronnin and Christopher Dance. Fisher kernels on visual vocabularies for image categorization. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

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.

John C Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in kernel methods*, pages 185–208. MIT press, 1999.

John C Platt, Nello Cristianini, and John Shawe-Taylor. Large margin dags for multiclass classification. In *nips*, volume 12, pages 547–553, 1999.

Zhaochun Ren, Maria-Hendrike Peetz, Shangsong Liang, Willemijn van Dolen, and Maarten de Rijke. Hierarchical multi-label classification of social text streams. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '14, pages 213–222, 2014.

Peter Richmond and Sorin Solomon. Power laws are disguised boltzmann laws. *International Journal of Modern Physics C*, 12(03):333–343, 2001.

Ryan Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *The Journal of Machine Learning Research*, 5:101–141, 2004.

Jorge Sánchez, Florent Perronnin, Thomas Mensink, and Jakob Verbeek. Image classification with the fisher vector: Theory and practice. *International journal of computer vision*, 105(3):222–245, 2013.

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

Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. 2002.

Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter. Pegasos: Primal estimated sub-gradient solver for svm. *Mathematical programming*, 127(1):3–30, 2011.

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

Herbert A Simon. On a class of skew distribution functions. *Biometrika*, 42(3/4): 425–440, 1955.

Chaoming Song, Shlomo Havlin, and Hernan A Makse. Self-similarity of complex networks. *Nature*, 433(7024):392–395, 2005.

Hideki Takayasu, Aki-Hiro Sato, and Misako Takayasu. Stable infinite variance fluctuations in randomly amplified langevin systems. *Physical Review Letters*, 79(6):966–969, 1997.

Lei Tang, Suju Rajan, and Vijay K. Narayanan. Large scale multi-label classification via metalabeler. In *18th international conference on WWW*, pages 211–220, 2009a.

Yuchun Tang, Yan-Qing Zhang, Nitesh V Chawla, and Sven Krasser. Svms modeling for highly imbalanced classification. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 39(1):281–288, 2009b.

Claudio J Tessone, Markus M Geipel, and Frank Schweitzer. Sustainable growth in complex networks. *EPL (Europhysics Letters)*, 96(5):58005, 2011.

Ambuj Tewari and Peter L Bartlett. On the consistency of multiclass classification methods. *The Journal of Machine Learning Research*, 8:1007–1025, 2007.

Vladimir Vapnik. *Statistical learning theory*. Wiley, 1998.

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.

Jason Weston. Multi-class support vector machines. *Technical Report CSD-TR-98-04*, 1998.

Jason Weston, Ameesh Makadia, and Hector Yee. Label partitioning for sublinear ranking. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 181–189, 2013.

Kenneth G Wilson and John Kogut. The renormalization group and the expansion. *Physics Reports*, 12(2):75–199, 1974.

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

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 Informaion Retrieval*, pages 619–626, 2008.

Jian-Bo Yang and Ivor W. Tsang. Hierarchical maximum margin learning for multi-class classification. In *UAI 2011, Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence, Barcelona, Spain, July 14-17, 2011*, pages 753–760, 2011.

Jian-Bo Yang and Ivor W Tsang. Hierarchical maximum margin learning for multi-class classification. *arXiv preprint arXiv:1202.3770*, 2012.

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 Informaion Retrieval*, pages 42–49. ACM, 1999.

Yiming Yang, Jian Zhang, and Bryan Kisiel. A scalability analysis of classifiers in text categorization. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, SIGIR '03, pages 96–103, 2003.

Hsiang-Fu Yu, Prateek Jain, and Inderjit S Dhillon. Large-scale multi-label learning with missing labels. *arXiv preprint arXiv:1307.5101*, 2013.

G Udny Yule. A mathematical theory of evolution, based on the conclusions of dr. jc willis, frs. *Philosophical Transactions of the Royal Society of London. Series B, Containing Papers of a Biological Character*, 213:21–87, 1925.

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.

Tong Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *Proceedings of the twenty-first international conference on Machine learning*, page 116. ACM, 2004a.

Tong Zhang. Statistical behavior and consistency of classification methods based on convex risk minimization. *Annals of Statistics*, pages 56–85, 2004b.