Using the Progressive Filtering Approach to Deal with Input Imbalance in Large-Scale Taxonomies

Andrea Addis, Giuliano Armano, and Eloisa Vargiu

DIEE, University of Cagliari, Italy,
{addis, armano, vargiu}@diee.unica.it

Abstract. Hierarchical text categorization is a challenging task to be investigated when adopting large-scale taxonomies. Real-world scenarios are typically characterized by a huge amount of non relevant documents with respect to the documents a user is looking for (the input imbalance problem). In this paper, we are interested in investigating how the ratio between relevant and non relevant documents affects the performances of a classifier system. To this end, we propose a progressive filtering approach. Experiments, performed on DMOZ, confirm the validity of the approach.

1 Introduction

Large-scale taxonomies, as web taxonomies (e.g., DMOZ[1]), are typically characterized by tens or hundreds of thousands of categories, deep hierarchies, and imbalanced category distribution over documents. As noted by [2] [30], due to the increasing importance of term polysemy for large corpora, both precision and recall decrease as the number of categories increases, so that considering categories organized in a hierarchy may help to improve the overall performance.

To this end, several hierarchical text categorization approaches have been investigated [14] [24] [25] [6] [10]. Furthermore, an open issue is still whether state-of-the-art approaches in text categorization can scale to large taxonomies without performing worse [19] [20] [28] [4].

Another important issue concerns the benefits that a hierarchical approach can give in real-world scenarios, typically characterized by imbalanced data [15]. In fact, in these scenarios, relevant and non relevant documents (i.e., positive and negative examples, respectively) are typically imbalanced, turning classifiers trained with the same percent of positive and negative examples into inadequate tools.

In this work, we are interested in studying a hierarchical text categorization approach, called Progressive Filtering (PF), focusing on the input imbalance that typically occurs in real-world scenarios. It is worth pointing out in advance that, in its simplest setting, PF decomposes a given rooted taxonomy into pipelines,

[1] http://www.dmoz.org

one for of each path that exists between the root and each node of the taxonomy, so that each pipeline can be tuned in isolation. In particular, a threshold selection algorithm (TSA) can be run to identify an optimal, or sub-optimal, combination of thresholds for each pipeline.

The rest of the paper is organized as follows: in Section 2, a brief overview of the input imbalance problem is given. Sections 3 describes the PF. In Section 4, TSA is presented. Experiments and results are illustrated in Section 5. Section 6 ends the paper with conclusions and future work.

2 The Input Imbalance Problem

Japkowicz [12] studied the class imbalance problem, i.e. the problem related to domains in which (in binary classification) one class is represented by a large number of examples whereas the other is represented by only a few. High imbalance occurs in real-world domains where the decision system is aimed at detecting a rare but important case [15]. The problem of imbalance has got more and more emphasis in recent years. Imbalanced data sets exist in many real-world domains, such as spotting unreliable telecommunication customers, detection of oil spills in satellite radar images, learning word pronunciations, text classification, detection of fraudulent telephone calls, information retrieval and filtering tasks, and so on [27] [29].

A number of solutions to the class imbalance problem have been proposed both at the data- and algorithmic-level. Data-level solutions [17] [7] [16] include many different forms of resampling such as random oversampling with replacement, random undersampling, directed oversampling, directed undersampling, oversampling with informed generation of new samples, and combinations of the above techniques. Algorithmic-level solutions include adjusting the costs of the various classes in order to counteract the class imbalance [22], adjusting the probabilistic estimate at the tree leaf [32], adjusting the decision threshold [26], and recognition-based [23] rather than discrimination-based learning [13]. Hybrid approaches, which combine existing ones, have been also used to handle the class imbalance problem [9] [11]. Furthermore, in [33] the authors suggest that existing measures used for feature selection are not very appropriate for imbalanced data sets. They propose a feature selection framework, which selects features for positive and negative classes separately and then explicitly combines them.

In this work, we are interested in studying how to cope with input imbalance adopting a hierarchical text categorization approach. The underlying motivation is that, in real world applications, the ratio between interesting and uninteresting documents is typically very low, so that classifiers trained with a balanced training set are inadequate to deal with those environments. To this end, we perform the training activity in two phases. First, each classifier is trained by using a balanced dataset, then, a threshold selection algorithm is applied and thresholds are calculated taking into account the input imbalance.

3 PF

Progressive Filtering (PF) unfolds the taxonomy into pipelines of classifiers, as depicted in Figure 1.

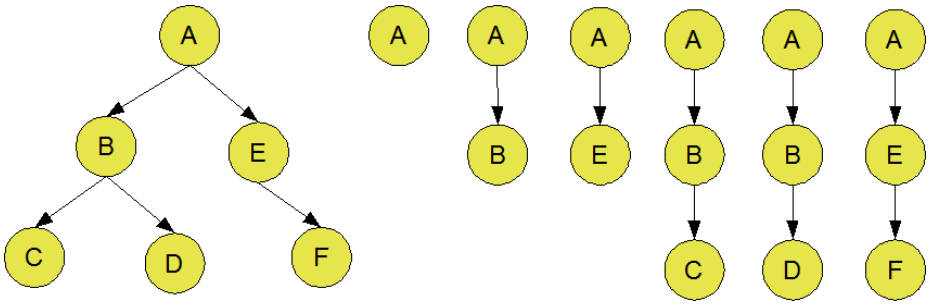

Fig. 1. A taxonomy and its corresponding pipelines.

Each node of the pipeline is a binary classifier able to recognize whether or not an input belongs to the corresponding class (i.e., to the corresponding node of the taxonomy).

As for document representation, we adopt the typical method for representing texts that uses the bag of words approach, in which each word from a vocabulary corresponds to a feature and a document to a feature vector. First, all non-informative words such as prepositions, conjunctions, pronouns and very common verbs are disregarded by using a stop-word list; subsequently, the most common morphological and inflexional suffixes are removed by adopting a standard stemming algorithm [21]. After having determined the overall sets of features, their values are computed for each document resorting to the well-known TF-IDF method. To reduce the high dimensionality of the feature space, we locally select the features that represent a node by adopting the information gain method [31].

As for the training activity, first, each classifier is trained by using a balanced dataset by adopting a hierarchical training set, according to [6]. For any given node, the training set contains documents taken from the subtree rooted in a category as positive examples, and documents of the sibling subtrees as negative examples. Figure 2 shows the hierarchical training set (2-a) in opposition to a proper training set (2-b).

Given a taxonomy, where each node represents a classifier entrusted with recognizing all corresponding positive inputs (i.e., interesting documents), each input traverses the taxonomy as a "token", starting from the root. If the current classifier recognizes the token as positive, it passes it on to all its children (if any), and so on. A typical result consists of activating one or more branches within the taxonomy, in which the corresponding classifiers have been activated by the given token. Let us note that, partitioning the taxonomy in pipelines

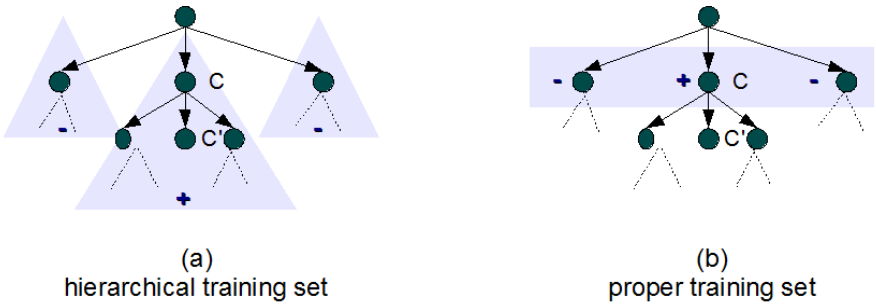

Fig. 2. Types of training sets [5].

gives rise to a set of new classifiers, each actually represented by a pipeline. For instance, the taxonomy depicted in Figure 1 gives rise to six pipelines.

4 TSA

It is worth pointing out in advance that the same classifier may have different behaviors, depending on which pipeline it is embedded. As a consequence, a relevant related problem is how to calibrate the threshold of the binary classifiers embedded by each pipeline in order to optimize the pipeline behavior. Furthermore, each pipeline is considered in isolation from the others. For instance, given $\pi_1 = \langle A, B, C\rangle$ and $\pi_2 = \langle A, B, D\rangle$, the classifiers A and B are not compelled to share the same threshold.

Given a set of documents D and a set of labels C, in classical text categorization a function $CSV_i : D \rightarrow [0, 1]$ exists for each $c_i \in C$. The behavior of c_i is controlled by a threshold θ_i, responsible for relaxing or restricting the acceptance rate of the corresponding classifier. In particular, given $d \in D$, $CSV_i(d) \geq \theta_i$ is interpreted as a decision to categorize d under c_i, whereas $CSV_i(d) < \theta_i$ is interpreted as a decision not to categorize d under c_i. In PF, let us still assume that CSV_i exists with the same semantics adopted in the classical case. Considering a pipeline π, composed by n classifiers, the acceptance policy strictly depends on the vector of thresholds $\Theta_\pi = \langle \theta_1, \theta_2, \cdots, \theta n\rangle$ that embodies the thresholds of all classifiers in π. In order to categorize d under π, the following constraint must be satisfied: for $k = 1..n$, $CSV_i(d) \geq \theta_k$. On the contrary, d is not categorized under c_i in the event that a classifier in π exists that rejects it.

Unfortunately, the task of threshold calibration is characterized by a high time complexity, being actually a problem of meta-learning (i.e., a learning problem whose *instances* are in fact learning problems themselves). Hence, a greedy algorithm for calibrating thresholds would be be very helpful in the task of finding a suboptimal solution. Given a utility function, we are interested in finding an effective (and computationally "light") way to reach a sub-optimum in the task of determining which are the best thresholds. To this end, for each pipeline

π a sub-optimal combination of thresholds is searched for, considering the actual ratio between positive and negative examples that depends on the given scenario.

Bearing in mind that the lower the threshold the less restrictive the classifier, the sub-optimal combination of thresholds is calculated according to the constraints imposed by TSA, a greedy bottom-up algorithm that relies on two functions:

- *repair* ($\mathfrak{R}$), which operates on a classifier C by increasing, $\mathfrak{R}(up, C)$, or decreasing, $\mathfrak{R}(down, C)$, its threshold until the selected utility function reaches a local maximum[2].
- *calibrate* ($\mathfrak{C}$), which operates going downwards from the given classifier to its offspring by repeatedly calling $\mathfrak{R}$. It is intrinsically recursive and at each step it calls $\mathfrak{R}$ to calibrate the current classifier.

Given a pipeline $\pi = \langle C_1, C_2, \cdots, C_L \rangle$, where L is its depth, first all thresholds are set to zero. TSA is then defined as follows:

$$TSA(\pi) := for\, k = L\, downto\, 1\, do\, \mathfrak{C}(up, C_k) \tag{1}$$

which indicates that $\mathfrak{C}$ is applied at each node of the pipeline, starting from the leaf ($k = L$).

The calibrate function is defined as follows:

$$\begin{aligned} \mathfrak{C}(up, C_k) &:= \mathfrak{R}(up, C_k),\ k = L \\ \mathfrak{C}(up, C_k) &:= \mathfrak{R}(up, C_k) + \mathfrak{C}(down, C_k),\ k < L \end{aligned} \tag{2}$$

and

$$\begin{aligned} \mathfrak{C}(down, C_k) &:= \mathfrak{R}(down, C_k),\ k = L \\ \mathfrak{C}(down, C_k) &:= \mathfrak{R}(down, C_k) + \mathfrak{C}(up, C_k),\ k < L \end{aligned} \tag{3}$$

where the + operator actually denotes a sequence operator, meaning that in the formula $a + b$ action a is performed *before* action b.

To better illustrate the approach, let us consider the unfolding reported in see Figure 3, which corresponds to the pipeline of three classifiers $\pi = \langle C_1, C_2, C_3 \rangle$:

$$\begin{aligned} TSA(\pi) := for\, k = 3\, downto\, 1\, do\, \mathfrak{C}(up, C_k) = \\ \mathfrak{C}(up, C_3) = \mathfrak{R}(up, C_3),\ \textbf{\textit{step 1}} \\ \mathfrak{C}(up, C_2) = \mathfrak{R}(up, C_2) + \mathfrak{C}(down, C_3) = \\ = \mathfrak{R}(up, C_2) + \mathfrak{R}(down, C_3),\ \textbf{\textit{step 2}} \\ \mathfrak{C}(up, C_1) = \mathfrak{R}(up, C_1) + \mathfrak{C}(down, C_2) = \\ = \mathfrak{R}(up, C_1) + \mathfrak{R}(down, C_2) + \mathfrak{C}(up, C_3) = \\ = \mathfrak{R}(up, C_1) + \mathfrak{R}(down, C_2) + \mathfrak{R}(up, C_3),\ \textbf{\textit{step 3}} \end{aligned} \tag{4}$$

Once calculated the sub-optimal combination of thresholds for a given imbalance, the pipelines are ready to be used in the corresponding scenario. Let us note,

[2] Different utility functions (e.g., precision, recall, F_β, user-defined) can be adopted, depending on the constraint imposed by the underlying scenario.

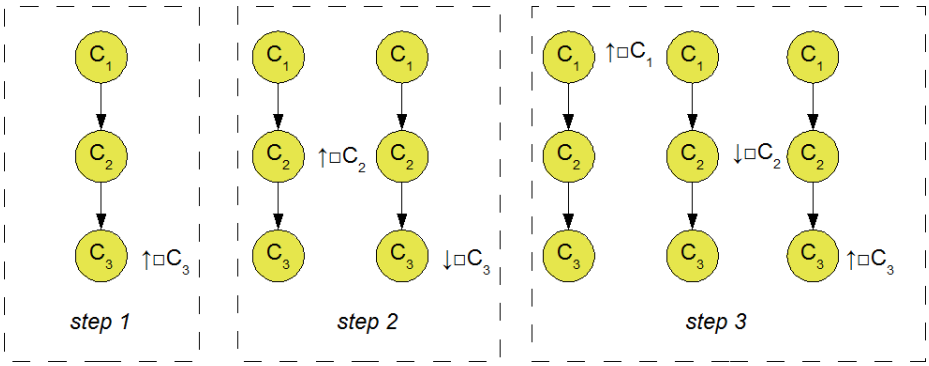

Fig. 3. Unfolding the threshold-selection procedure for a pipeline composed by three classifiers.

here, that this combination depends on both the adopted dataset and the actual input imbalance. In fact, as noted in [18], different goals for the system lead to different optimal behaviors.

5 Experiments and Results

Experiments have been performed to validate the proposed approach with respect to the impact of PF on the input imbalance. To this end, three series of experiments have been performed: first, performances calculated using PF have been compared with those calculated by using to the corresponding flat approach. Then, PF has been tested to assess the improvement of performances while augmenting the pipeline depth. Finally, according to [6], performances have been calculated in terms of generalization-, specialization-, misclassification-, and unknown-error.

Experiments have been performed by customizing to this specific task X.MAS [1], a generic multiagent architecture built upon JADE [3], devised to make it easier the implementation of information retrieval and information filtering applications.

The corpus chosen for this study is the benchmark dataset DMOZ, a collection of HTML documents referenced in a web directory developed in the Open Directory Project (ODP). To be able to explore the effects of the taxonomomy depth on PF performances, we selected 17 categories with a maximum depth of 5. The underlying motivation is that in this way each category have at least 300 documents for different test phases. The corresponding training set includes a subset of the 136,000 documents, each one belonging to one category.

The main question we are interested in is the effectiveness of the proposed approach with respect to flat classification. In order to make a fair comparison, the same classification technique has been adopted, i.e., *wk*-NN [8]. The motivation for the adoption of this particular technique stems from the fact that it does not require specific training and is very robust with respect to noisy data.

First, each classifier is trained with a balanced data set of 100 documents by using 200 (TFIDF) features selected in accordance with their information gain. Then, the best thresholds are selected. To make a fair comparison, both the thresholds of the pipelines and the flat classifiers have been chosen considering the F_1 measure as utility function. As for the pipelines, TSA described in Section 4 has been adopted. During the calibration / repair phase a step of 10^{-4} has been used by TSA to increase (decrease) thresholds.

Experiments have been performed by assessing the behavior of the proposed hierarchical approach in presence of different percentage of positive examples versus negative examples, i.e., from 2^{-1} to 2^{-7}. To better put into evidence the importance of the pipeline depth, for PF we considered only pipelines that end with a leaf node of the taxonomy. Accordingly, for the flat approach, we considered only classifiers that correspond to a leaf.

PF vs Flat Classification. To assess the effectiveness of the approach with respect to the corresponding flat approach, we calculated macro-averaging precision and recall. Figures 4 and 5 show macro-averaging of precision and recall, respectively. Precision and recall have been calculated for both the flat classifiers and the pipelines by varying the input imbalance. As pointed out by experimental results (for precision), the distributed solution based on pipelines has reported better results than those obtained with the flat model. On the contrary results on recall are worse than those obtained with the flat model.

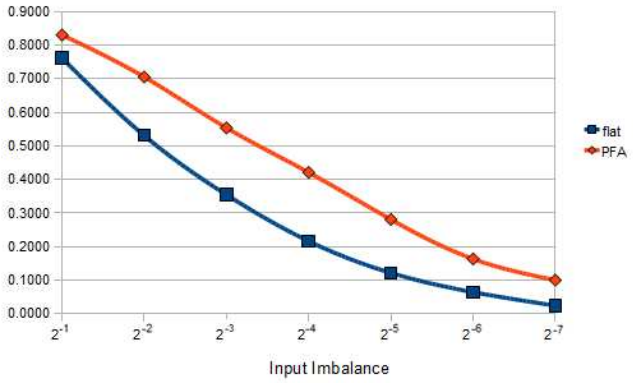

Fig. 4. Comparison of precision

Performance improvement. The improvement has been calculated in terms of F_1, for different values of imbalance between positive and negative examples and for pipelines of different lengths, i.e., varying the positive ratio by

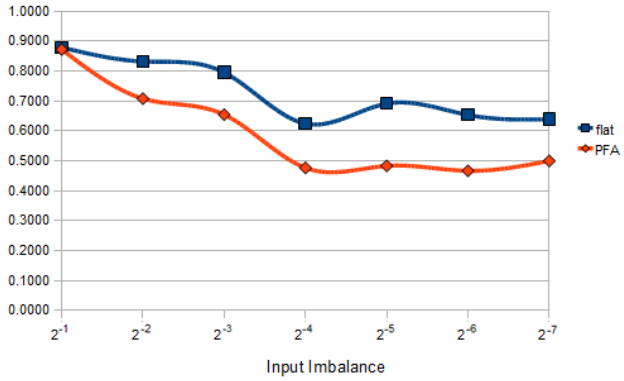

Fig. 5. Comparison of recall

a power of two. The performance improvement is then calculated in percentage with the formula $(F_1(pipeline) - F_1(flat))\%$.

Figure 6 shows the performance improvement augmenting the pipeline depth. Experimental results –having the adopted taxonomy a maximum depth of five– show that PF performs always better than the flat approach, the filtering effect of a pipeline being not negligible. Nevertheless, let us note that these results are not always statistically significant, depending on the amount of examples of the categories under analysis. This is an unavoidable side effect related with the decision of making experiments with pipelines of length five.

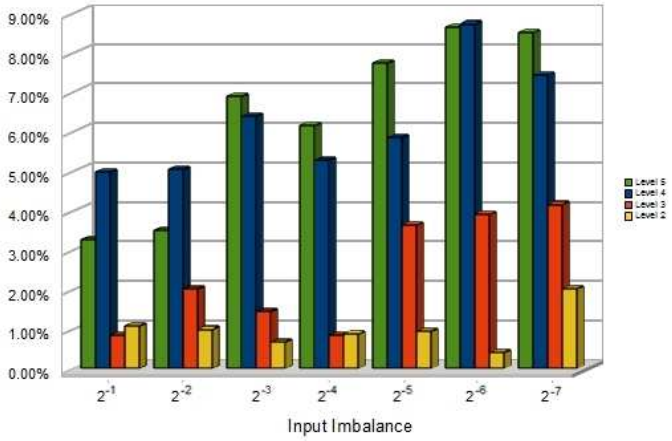

Fig. 6. Performance improvement

Hierarchical Metrics. According to [6], we evaluate the overall error in terms of generalization-, specialization-, misclassification-, and unknown-error. For the sake of completeness, let us briefly recall here the adopted four hierarchical metrics:

- misclassification error, i.e., the percentage of documents misclassified into a category not related to the correct category in the hierarchy;
- generalization error, i.e., the percentage of documents misclassified into a supercategory of the correct one;
- specialization error, i.e., the percentage of documents misclassified into a subcategory of the correct one;
- unknown ratio, i.e., the percentage of rejected documents.

Let us note that this results strictly depend on the adopted utility function. In fact, changing the utility function, the confusion matrix of the pipeline changes accordingly.

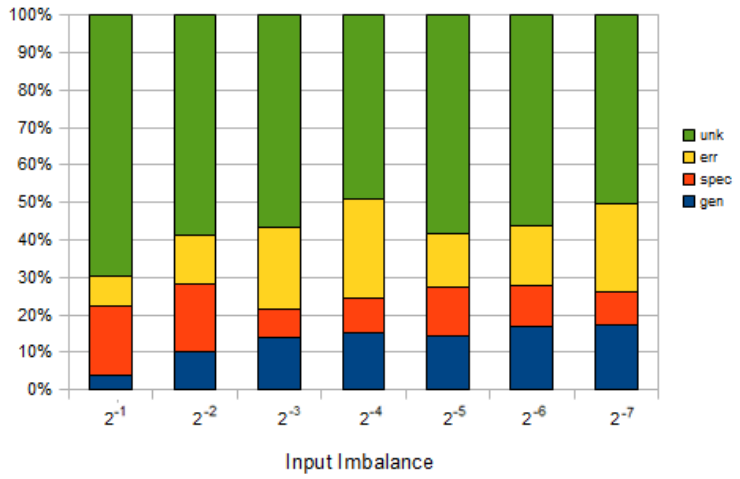

Fig. 7. Hierarchical measures

Figure 7 depicts the results obtained varying the imbalance. Analyzing the results, it is easy to note that the generalization- and the misclassification-error grow with the imbalance, whereas the specialization-error and the unknown-ration decrease. As for the generalization-error, i.e., the percentage of documents misclassified in a supercategory of the correct one, it depends on the overall number of FNs. The greater the imbalance the greater the amount of FNs; hence the generalization-error increases with the imbalance. In presence of input imbalance, the trend of the generalization-error is similar to the trend of the recall. As for the specialization-error, i.e., the percentage of documents misclassified in a subcategory of the correct one, it depends on the overall number of FPs. The

greater the imbalance, the lower the amount of FPs, so that the specialization-error decreases with the imbalance. In presence of input imbalance, the trend of the specialization-error is similar to the trend of the precision. As for the unknown-ratio and misclassification-error, let us point out that an imbalance between positive and negative examples can be suitably dealt with by exploiting the filtering effect of classifiers in the pipeline.

As a final remark, let us note that, in order to improve the performance (or to favour a specific metric), a suitable utility function can be adopted. In fact, we use F_1 that weight in same manner precision and recall. A different utility function could be used, for example, to reduce the overall number of FNs accepting as positive documents that are actually TNs. In this way, the unknown-ratio (which depends on the amount of FNs) decreases, whereas the misclassification-error (which depends on the amount of FPs) increases.

6 Conclusions and Future Work

In this paper, we made experiments the with PF to investigate how the ratio between positive and negative examples affects the performances of a categorization system and how these performances can be improved by adopting PF instead of a classical flat approach. Results show that PF is able to counteract an imbalance of up to 100 negative examples vs. 1 positive example.

As for the future work, several issues are currently under investigation. From a research perspective, the whole taxonomy could be investigate instead of the corresponding set of pipelines. From a benchmarking perspective, further metrics to assess the performances of PF could be adopted and calculated. From an experimental perspective, PF could be tested on other datasets, such as Reuters, TREC or MeSH.

References

1. A. Addis, G. Armano, and E. Vargiu. From a generic multiagent architecture to multiagent information retrieval systems. In *AT2AI-6, Sixth International Workshop, From Agent Theory to Agent Implementation*, pages 3–9, 2008.
2. C. Apté, F. Damerau, and S. M. Weiss. Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems*, 12(3):233–251, 1994.
3. F. Bellifemine, G. Caire, and D. Greenwood. *Developing Multi-Agent Systems with JADE (Wiley Series in Agent Technology)*. John Wiley and Sons, 2007.
4. P. N. Bennett and N. Nguyen. Refined experts: improving classification in large taxonomies. In *SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 11–18, New York, NY, USA, 2009. ACM.
5. M. Ceci and D. Malerba. Hierarchical classification of HTML documents with WebClassII. In F. Sebastiani, editor, *Proceedings of ECIR-03, 25th European Conference on Information Retrieval*, pages 57–72. Berlin Heidelberg NewYork: Springer, 2003.

6. M. Ceci and D. Malerba. Classifying web documents in a hierarchy of categories: a comprehensive study. *Journal of Intelligent Information Systems*, 28(1):37–78, 2007.
7. N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.
8. W. Cost and S. Salzberg. A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning*, 10:57–78, 1993.
9. A. Estabrooks and N. Japkowicz. A mixture-of-experts framework for learning from imbalanced data sets. In *IDA '01: Proceedings of the 4th International Conference on Advances in Intelligent Data Analysis*, pages 34–43, London, UK, 2001. Springer-Verlag.
10. A. Esuli, T. Fagni, and F. Sebastiani. Boosting multi-label hierarchical text categorization. *Inf. Retr.*, 11(4):287–313, 2008.
11. H. Guo and H. L. Viktor. Learning from imbalanced data sets with boosting and data generation: the databoost-im approach. *SIGKDD Explor. Newsl.*, 6(1):30–39, 2004.
12. N. Japkowicz. Learning from imbalanced data sets: a comparison of various strategies. In *AAAI Workshop on Learning from Imbalanced Data Sets*, 2000.
13. N. Japkowicz. Concept-learning in the presence of between-class and within-class imbalances. In *AI '01: Proceedings of the 14th Biennial Conference of the Canadian Society on Computational Studies of Intelligence*, pages 67–77, London, UK, 2001. Springer-Verlag.
14. D. Koller and M. Sahami. Hierarchically classifying documents using very few words. In D. H. Fisher, editor, *Proceedings of ICML-97, 14th International Conference on Machine Learning*, pages 170–178, Nashville, US, 1997. Morgan Kaufmann Publishers, San Francisco, US.
15. S. Kotsiantis, D. Kanellopoulos, and P. Pintelas. Handling imbalanced datasets: a review. *GESTS International Transactions on Computer Science and Engineering*, 30.
16. S. Kotsiantis and P. Pintelas. Mixture of expert agents for handling imbalanced data sets. *Ann Math Comput Teleinformatics*, 1:46–55, 2003.
17. M. Kubat and S. Matwin. Addressing the curse of imbalanced training sets: One-sided selection. In *In Proceedings of the Fourteenth International Conference on Machine Learning*, pages 179–186. Morgan Kaufmann, 1997.
18. D. D. Lewis. Evaluating and optimizing autonomous text classification systems. In *SIGIR '95: Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 246–254, New York, NY, USA, 1995. ACM.
19. T.-Y. Liu, Y. Yang, H. Wan, H.-J. Zeng, Z. Chen, and W.-Y. Ma. Support vector machines classification with a very large-scale taxonomy. *SIGKDD Explor. Newsl.*, 7(1):36–43, 2005.
20. T.-Y. Liu, Y. Yang, H. Wan, Q. Zhou, B. Gao, H.-J. Zeng, Z. Chen, and W.-Y. Ma. An experimental study on large-scale web categorization. In *WWW '05: Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 1106–1107, New York, NY, USA, 2005. ACM.
21. M. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.
22. F. Provost and T. Fawcett. Robust classification for imprecise environments. *Machine Learning*, 42(3):203–231, 2001.
23. B. Raskutti and A. Kowalczyk. Extreme re-balancing for svms: a case study. *SIGKDD Explor. Newsl.*, 6(1):60–69, 2004.

24. M. E. Ruiz and P. Srinivasan. Hierarchical text categorization using neural networks. *Information Retrieval*, 5(1):87–118, 2002.
25. A. S. Weigend, E. D. Wiener, and J. O. Pedersen. Exploiting hierarchy in text categorization. *Information Retrieval*, 1(3):193–216, 1999.
26. G. M. Weiss. Mining with rarity: a unifying framework. *SIGKDD Explor. Newsl.*, 6(1):7–19, 2004.
27. G. Wu and E. Y. Chang. Class-boundary alignment for imbalanced dataset learning. In *In ICML 2003 Workshop on Learning from Imbalanced Data Sets*, pages 49–56, 2003.
28. G.-R. Xue, D. Xing, Q. Yang, and Y. Yu. Deep classification in large-scale text hierarchies. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 619–626, New York, NY, USA, 2008. ACM.
29. R. Yan, Y. Liu, R. Jin, and A. Hauptmann. On predicting rare classes with svm ensembles in scene classification. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03). 2003 IEEE International Conference on*, volume 3, pages III–21–4 vol.3, April 2003.
30. Y. Yang. An evaluation of statistical approaches to medline indexing. In *Proceedings of the American Medical Informatic Association (AMIA)*, pages 358–362, 1996.
31. Y. Yang and J. O. Pedersen. Feature selection in statistical learning of text categorization. In *Proceedings of the 14th International Conference of Machine Learning ICML9*, pages 412–420, 1997.
32. S.-J. Yen and Y.-S. Lee. Cluster-based under-sampling approaches for imbalanced data distributions. *Expert System Applications*, 36(3):5718–5727, 2009.
33. Z. Zheng, X. Wu, and R. Srihari. Feature selection for text categorization on imbalanced data. *SIGKDD Explor. Newsl.*, 6(1):80–89, 2004.