

Adaptive selection of base classifiers in one-against-all learning for large multi-labeled collections

Arturo Montejo Ráez¹, Luís Alfonso Ureña López² and Ralf Steinberger³

¹ European Laboratory for Nuclear Research, Geneva, Switzerland

² Department of Computer Science, University of Jaén, Spain

³ European Commission, Joint Research Centre, Ispra, Italy

Abstract. In this paper we present the problem found when studying an automated text categorization system for a collection of High Energy Physics (HEP) papers, which shows a very large number of possible classes (over 1,000) with highly imbalanced distribution. The collection is introduced to the scientific community and its imbalance is studied applying a new indicator: the *inner imbalance degree*. The one-against-all approach is used to perform multi-label assignment using Support Vector Machines. Over-weighting of positive samples and S-Cut thresholding is compared to an approach to automatically select a classifier for each class from a set of candidates. We also found that it is possible to reduce computational cost of the classification task by discarding classes for which classifiers cannot be trained successfully.

1 Introduction

The automatic assignment of keywords to documents using full-text data is a subtask of *Text Categorization*, a growing area where Information Retrieval techniques and Machine Learning algorithms meet offering solutions to problems with real world collections.

We can distinguish three paradigms in text categorization: the *binary* case, the *multi-class* case and the *multi-label* case. In the binary case a sample either belongs or does not belong to one of two given classes. In the multi-class case a sample belongs to just one of a set of m classes. Finally, in the multi-label case, a sample may belong to several classes at the same time, that is, classes are *overlapped*. In binary classification a classifier is trained, by means of supervised algorithms, to assign a sample document to one of two possible sets. These sets are usually referred to as belonging (positive) or not belonging (negative) samples respectively (the one-against-all approach), or to two disjoint classes (the one-against-one approach). For these two binary classification tasks we can select among a wide range of algorithms, including Naïve Bayes, Linear Regression, Support Vector Machines (SVM) [8] and LVQ [11]. SVM has been reported to outperform the other algorithms. The binary case has been set as a base case from which the two other cases are derived. In multi-class and multi-label assignment,

the traditional approach consists of training a binary classifier for each class, and then, whenever the binary base case returns a measure of confidence on the classification, assigning either the top ranked one (multi-class assignment) or a given number of the top ranked ones (multi-label assignment). More details about these three paradigms can be found in [1]). We will refer to the ranking approach as the *battery* strategy because inter-dependency is not taken into consideration.

Another approach for multi-labeling consists of returning all those classes whose binary classifiers provide a positive answer for the sample. It has the advantage of allowing different binary classifiers for each class, since inter-class scores do not need to be coherent (since there is no ranking afterwards). Better results have been reported when applying one-against-one in multi-class classification [1], but in our multi-label case this is not an option because any class could theoretically appear together with any other class, making it difficult to establish disjoint assignments. This is the reason why one-against-all deserves our attention in the present work.

Although classification is subject to intense research (see [18]), some issues demand more attention than they have been given so far. In particular, problems relating to *multi-label* classification would require more attention. However, due to the lack of available resources (mainly multi-labeled document collections), this area advances more slowly than others. Furthermore, multi-label assignment should not simply be studied as a general multi-class problem (which itself is rather different from the binary case), but it needs to be considered as a special case with additional requirements. For instance, in multi-label cases, some classes are inter-related, the degree of imbalance is usually radically different from one class to the next and, from a performance point of view, the need of comparing a sample to every single classifier is a waste of resources.

2 The class imbalance problem

Usually, multi-labeled collections make use of a wide variety of classes, resulting in an unequal distribution of classes throughout the collection and a high number of rare classes. This means not only that there is a strong imbalance between positive and negative samples, but also that some classes are used much more frequently than other classes. This phenomenon, known as the *class imbalance problem*, is especially relevant for algorithms like the C4.5 classification tree [4, 3] and margin-based classifiers like SVM [16, 20, 7].

Extensive studies have been carried out on this subject as reported by Japkowicz [7], identifying three major issues in the class imbalance problem: *concept complexity*, *training set size* and *degree of imbalance*. Concept complexity refers to the degree of “sparsity” of a certain class in the feature space (the space where document vectors are represented). This means that a hypothetical clustering algorithm acting on a class with high concept complexity would establish many small clusters for the same class. Regarding the second issue, i.e. the lack of a significantly large training sets, the only possible remedy is the usage of

over-sampling when the amount of available samples is insufficient, and under-sampling techniques for classes with too many samples, e.g. just using a limited number of samples for training a SVM, by selecting those positive and negative samples that are close to each other in the feature space. The validity of these techniques is also subject to debate [4]. Finally, Japkowicz defines the degree of imbalance as an index to indicate how much a class is more represented over another, including both the degree of imbalance between classes (what we call *inter-class imbalance*) and between its positive and negative samples (what we call the *inner imbalance degree*). Unfortunately, Japkowicz defined these values for her work towards the generation of an artificial collection and rewrote them later to fit specific problems regarding fixed parameters and the C5.0 algorithm, which make them difficult to manipulate. For these reasons, we cannot reuse her equations and propose here a variant focusing on the multi-label case.

We define the *inner imbalance degree* of a certain class i as a measure of the positive samples over the total of samples:

$$i_i = |1 - 2n_i/n| \tag{1}$$

where

n is the total number of samples and

n_i is the total number of samples having the class i in their labels.

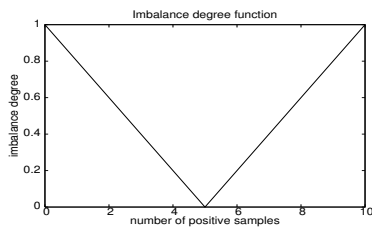


Fig. 1. The linear 'imbalance degree' function

Japkowicz' definition of imbalance degree helps in the generation of artificial distributions of documents to classes. Its value does not lie within a defined range, which makes it difficult to manipulate and compare with the degree of other classes in different partitions. The value proposed in equation 1 is zero for perfectly balanced classes, i.e. when the number of positive and negative samples are the same. It has a value of 1 when all samples are either positive or negative for that class. Its linear behavior is shown in figure 1 and, as we can see, it varies within the range [0,1].

3 The HEP collection

A very suitable document set for multi-label categorization research is the HEP collection of preprints, available from the European Laboratory for Nuclear Research. Some experiments have been carried out using this collection ([13, 12]), and its interesting distribution of classes allows us to carry out a number of experiments and to design a new approach. An analysis of the collection has shown that there is the typical high level of imbalance among classes. If a given class is rarely represented in a collection, we can intuitively foresee a biased training

that will yield classifiers with a low performance. It is clear that, if the collection were perfectly balanced, we could expect better categorization results, due to better learning.

The `hep-ex` partition of the HEP collection is composed of 2802 abstracts related to experimental high-energy physics that are indexed with 1093 main keywords (the categories).⁴ Figure 2 shows the distribution of keywords across the collection.

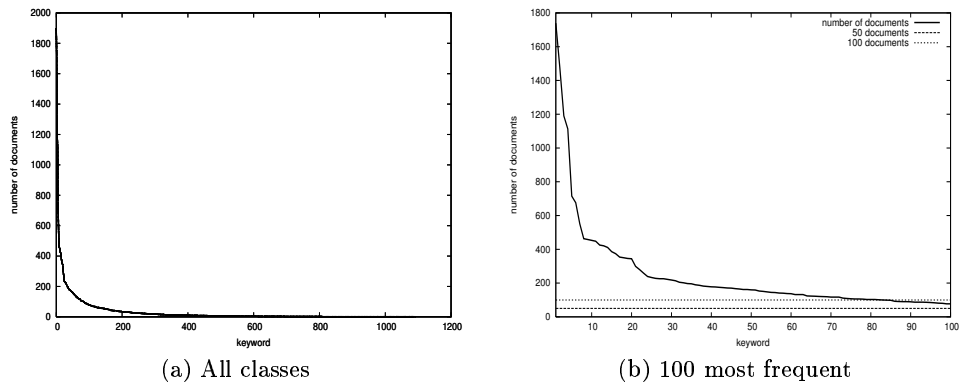


Fig. 2. Distribution of classes across documents in the `hep-ex` partition.

As we can see, this partition is **very** imbalanced: only 84 classes are represented by more than 100 samples and only five classes by more than 1000. The uneven use is particularly noticeable for the ten most frequent keywords: In table 1 the left column shows the number of positive samples of a keyword and the right column shows the percentage over the total of samples in the collection.

No. docs.	Keyword
1898 (67%)	electron positron
1739 (62%)	experimental results
1478 (52%)	magnetic detector
1190 (42%)	quark
1113 (39%)	talk
715 (25%)	Z0
676 (24%)	anti-p p
551 (19%)	neutrino
463 (16%)	W
458 (16%)	jet

We can now study this collection applying the inner imbalance degree measure defined

Table 1. The ten most frequent main keywords in the `hep-ex` partition

⁴ We did not consider the keywords related to reaction and energy because they are based on formulae and other specific data that is not easily identifiable in the plain-text version of a paper.

in equation 1. The two graphs in figures 3a and 3b show the inner imbalance degree for the main keywords in the `hep-ex` partition. We can notice how fast the imbalance grows to a total imbalance degree of almost 1. When looking at the ten most frequent classes, we can see the effect of our degree estimation: classes 0 and 1 are more imbalanced than class 2, which gets the lowest degree of imbalance in the whole set of classes. It is due to the fact that, as shown by table 1, this class has almost the same number of positive and negative samples. From class 3 onwards, the imbalance then grows dramatically.

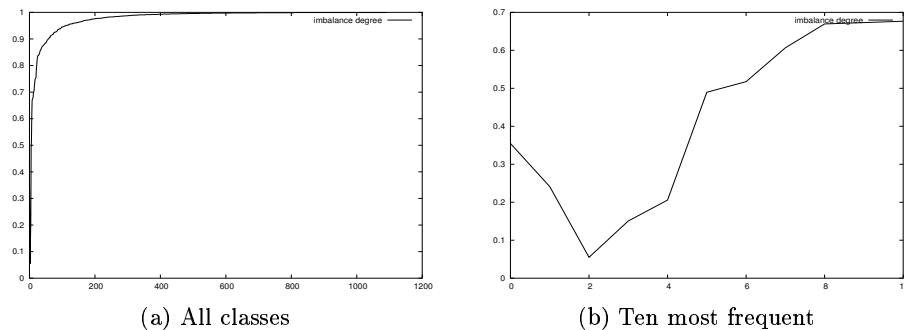


Fig. 3. Imbalance degree of classes in the `hep-ex` partition

When training binary classifiers for these keywords, we realized that the performance decreases strongly with growing imbalance degree. To correct document distribution across classes, we can use over-sampling (or under-sampling) or tune our classifiers accordingly. For example, for SVM we can set a cost factor, by which training errors on positive samples out-weights errors on negative samples [14]. We will use this in our experiments.

4 Balance weighting and classifier filtering

Some algorithms work better when, in the one-against-all approach, the number of positive samples is similar to the number of negative ones, i.e. when the class is balanced across the collection. However, multi-label collections are typically highly imbalanced. This is true for the HEP collection, but also for other known document sets like the OHSUMED medical collection used in the filtering track of TREC [5], and for the document collection of the European Institutions classified according to the EUROVOC thesaurus. This latter collection has been studied extensively for automatic indexing by Pouliquen et. al. (e.g. [2]), who exploit a variety of parameters in their attempt to determine whether some terms refer to one class or to another in the multi-labeled set.

The question now is how to deal with these collections when trying to apply binary learners that are sensitive to high imbalance degrees. We can use tech-

niques like over-sampling and under-sampling, as pointed out earlier, but this would lead to an overload of non-informational samples in the former case, and to the loss of information in the second case. Furthermore, concept complexity has also its effects on binary classifiers. We have not paid attention to this fact since it is out of the scope of the present study, but we should consider this to be yet another drawback for collections indexed with a large number of non-balanced classes.

In our experiments we basically train a system using the battery strategy, but (a), we allow tuning the binary classifier for a given class by a balance factor, and (b) we provide the possibility of choosing the best of a given set of binary classifiers. At CERN, we intend to apply our classification system to *real-time* environments so that a gain in classification speed is very important. Therefore, we have introduced a parameter α in the algorithm, resulting in the updated version given in figure 4. This value is a threshold for the minimum performance of a binary classifier during the validation phase in the learning process. If the performance of a certain classifier is below the value α , meaning that the classifier performs badly, we discard the classifier and the class completely. By doing this, we may decrease the recall slightly (since less classes get trained and assigned), but the advantages of increased computational performance and of higher precision compensate for it. The effect is similar to that of the *SCutFBR* proposed by Yang [21]. We never attempt to return a positive answer for rare classes. In the following, we show how this filtering saves us considering many classes without significant loss in performance.

We allow over-weighted positive samples using the actual fraction of positive samples over negative ones, that is, the weight for positive samples (w_+) is:

$$w_+ = C_- / C_+ \quad (2)$$

where

C_- is the total number of negative samples for the class
 C_+ is the total number of positive samples for the class

As we can see, the more positive documents we have for a given class, the lower the over-weight is, which makes sense in order to give more weight only when few positive samples are found. This method was used by Morik et al. [14] but they did not report how much it improved the performance of the classifier over the non-weighted scheme. As we said, this w_+ factor was used in our experiments to over-weight positive samples over negative ones, i.e. the classification error on a positive sample is higher than that of a negative one.

We also considered the *S-Cut* approach. The assignation of a sample as positive can be tuned by specifying the decision border. By default it is zero, but it can be set using the S-Cut algorithm [21]. This algorithm uses as threshold the one that gives the best performance on an evaluation set. That is, once the classifier has been trained, we apply it against an evaluation set using as possible thresholds the classification values (the margin for SVM). The threshold that reported the best performance (the highest F1 in our case) will be used.

Input:

- a set of multi-labeled training documents D_t
- a set of validation documents D_v
- a threshold α on the evaluation measure
- a set of possible label (classes) L ,
- a set of candidate binary classifiers C

Output :

- a set $C' = \{c_1, \dots, c_k, \dots, c_{|L|}\}$ of trained binary classifiers

Pseudo code:

```

 $C' = \emptyset$ 
for-each  $l_i$  in  $L$  do
   $T = \emptyset$ 
  for-each  $c_j$  in  $C$  do
     $train-classifier(c_j, l_i, D_t)$ 
     $T = T \cup \{c_j\}$ 
  end-for-each
   $c_{best} = best-classifier(T, D_v)$ 
  if  $evaluate-classifier(c_{best}) > \alpha$ 
     $C' = C' \cup \{c_{best}\}$ 
  end-if
end-for-each

```

Fig. 4. The one-against-all learning algorithm with classifier filtering

5 Experiments and results

5.1 Data preparation

The collection consists of 2967 full-text abstracts linked to 1103 main keywords. Each abstract was processed as follows:

- Punctuation was removed
- Every character was lowercased
- Stop words were removed
- The Porter stemming algorithm [15] was applied
- Resulting stems were weighted according to the TF.IDF scheme [17]

After processing the collection in this way, we trained the system applying each strategy using the *SVM-Light*⁵ package as the base binary classifier. We also filtered out classes not appearing in any document either in the training, validation or test sets, reducing the number of classes to 443.8 on average. Results are shown at the end of this section.

For the **evaluation** in experiments, *ten-fold cross validation* [9] was used in order to produce statistically relevant results that do not depend on the

⁵ SVM-Light is available at <http://svmlight.joachims.org/>

partitioning of the collection into training, validation and test sets. Extensive experiments have shown that this is the best choice to get an accurate estimate. The measures computed are *precision* and *recall*. The F_1 measure (introduced by Rijsbergen [19] a long time ago) is used as an overall indicator based on the two former ones and is the reference when filtering is applied. Also *accuracy* and *error* measurements are given for later discussion. The final values are computed using macro-averaging on a per-document basis, rather than the usual micro-averaging over classes. The reason is, again, the high imbalance in the collection. If we average by class, rare classes will influence the result as much as the most frequent ones, which will not provide a good estimate of the performance of the multi-label classifier over documents. Since the goal of this system is to be used for automated classification of individual documents, we consider it to be far more useful to concentrate on these measurements for our evaluation of the system. More details about these concepts can be found in [18], [10] and [22].

5.2 Results

Experiment	Precision	Recall	F1	Accuracy	Error	% of classes covered
No weight	74.07	33.96	43.92	98.23	1.77	33.96
No weight / Scut	74.26	34.44	44.38	98.24	1.76	99.95
Overweight 20	51.47	45.84	46.50	97.71	2.29	57.32
Auto weight	58.10	44.39	48.09	97.94	2.06	58.09
Overw. 2,5,10,20 / Scut	71.74	39.92	48.47	98.25	1.75	100.00
Auto weight / Scut	58.03	45.30	48.56	97.89	2.11	99.82
Overweight 2	70.74	40.45	48.78	98.21	1.79	53.36
Overweight 5	64.56	43.57	49.40	98.11	1.89	57.19
Overweight 10	62.30	45.22	50.14	98.08	1.92	57.30
Overw. 2,5,10,20	65.89	44.59	50.53	98.17	1.83	57.53

Table 2. Results of experiments using SVM

Table 2 shows the results of ten runs of our multi-label classifier with different configurations. The highest values of F_1 are reached when letting the system choose among fixed values for over-weighting positive samples (2, 5, 10 and 20). These are the results when applying the algorithm of figure 4 with $\alpha = 0.0$, i.e. no filtering over classifiers is done.

We see that the top recall reached does not imply having more classes trained. Therefore we may want to study how we can reduce the number of classes trained to speed up the classification process without losing too much in performance. For that purpose, we experimented with different values of α , as shown in tables 3 and 4.

α	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7
Precision	65.89	70.04	70.41	70.88	71.90	71.96	71.02	67.96
Recall	44.59	44.49	43.95	42.95	40.54	36.65	31.80	23.02
F_1	50.53	51.59	51.32	50.77	49.21	46.11	41.70	32.83
Accuracy	98.17	98.25	98.25	98.25	98.24	98.21	98.15	98.03
Error	1.83	1.75	1.75	1.75	1.76	1.79	1.85	1.97
% classes trained	57.53	56.49	50.81	43.20	32.73	23.23	16.00	8.58

Table 3. Results of experiments using multi-weighted SVM with filtering

α	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7
Precision	58.03	62.47	64.84	67.45	69.47	71.19	71.14	68.24
Recall	45.30	45.04	44.83	44.24	42.76	39.59	34.43	24.88
F_1	48.56	49.93	50.47	50.75	50.27	48.37	44.10	34.76
Accuracy	97.89	98.06	98.14	98.20	98.23	98.22	98.17	98.05
Error	2.11	1.94	1.86	1.80	1.77	1.78	1.83	1.95
% classes trained	99.82	85.30	77.10	68.47	55.74	42.34	30.82	16.72

Table 4. Results of experiments using auto-weighted S-Cut thresholded SVM with filtering

5.3 Analysis

Interesting conclusions can be drawn from the tables above. The first thing we notice is that recall is low compared to precision. This is normal if we consider the existence of rare and, therefore, difficult-to-train classes. When tuning our multi-label classifier, we see that variations in precision are more representative than for recall. The F_1 measure remains quite stable: throughout all the experiments with different configurations, the most we gain is 6.61%. However, a very important result is that, even when some configurations are able to train up to 100% of the total of classes involved (we can see how the percentage of classes successfully trained varies widely), it does not influence that much the overall performance of the classifier. We can conclude that *rare classes are not worth training*. This is the reason for the design of our filtering algorithm. Furthermore, it is not clear that S-Cut and auto-weighting strategies are so relevant for our data. As we can also notice, accuracy and error are not very sensitive to the variations of our parameters, but this is again due to imbalance: most of the classes are rare and for the most frequent ones we get high precision and recall, even with not very sophisticated configurations.

When discarding classes, we obviously gain in precision and, despite more classes not being trained, we do not lose that much in recall. The result is a better F_1 than without discarding, as shown by F_1 values in 2 compared to those of tables 3 and 4. We can see how strongly we can reduce the number of classes without affecting significantly the overall performance of the multi-label classifier. Figures 5a and 5b visualize the behavior described. The bigger our α

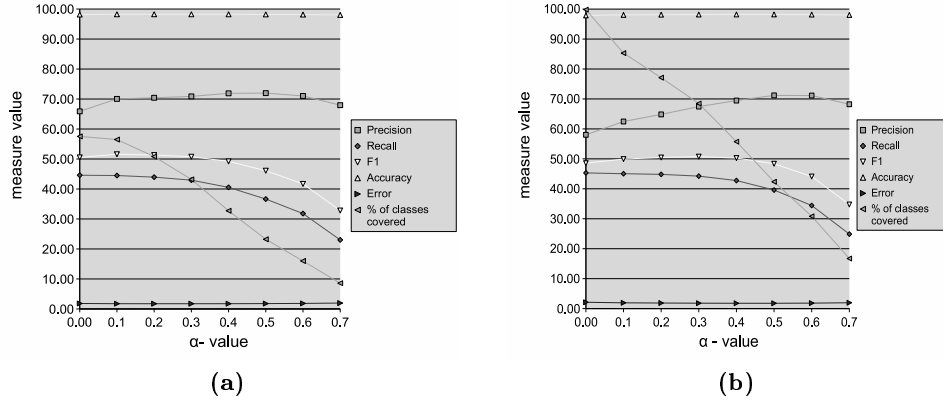


Fig. 5. Influence of filtering on (a) multi-weighted SVM and (b) auto-weighted with S-cut thresholding

is, the more classes are discarded. From all the test runs, the best value of F_1 was obtained with an α value of 0.1 and using candidate classifiers with over-weights 2, 5, 10 and 20 for positive classes. From the graphs we can see that increasing α yields to a higher precision up to a maximum from which the threshold will be so restrictive that even good classifiers are discarding and, therefore, the precision starts to decrease accordingly. Thus, our choice of α will depend on our preference of precision over recall and our need of reducing classes for faster classification. If we are able to discard non-relevant (rarely used) classes, we can almost maintain our performance classifying against a lower number of classes.

6 Conclusions and future work

We have presented a new collection for multi-label indexing. The `hep-ex` partition can be obtained by contacting the authors. A calculus for measuring the imbalance degree has been proposed, along with a study of the overweight of positive classes on this collection using SVM and the application of S-Cut. The results show that this is a relevant issue, and that an imbalance study of any multi-label collection should be carried out in order to properly select the base binary classifiers. Another promising issue would be to work on other aspects of imbalance like *concept complexity* [6]. We have started investigating this topic by working with “concepts” rather than with terms in order to reduce the term space. By doing this, we would cover the main drawbacks of imbalanced collections.

Filtering by classification thresholding is very effective to reduce the number of classes involved in multi-label classification. Without forcing expensive tuning of the threshold, we propose to provide a range of α values and let the algorithm choose the classifier with the best behavior.

One of the disadvantages using the battery approach is its computational cost, since we have to launch every classifier for a sample. However, SVM is quite selective, not being trainable in many cases, discarding in this way many conflictive classes. This reduces the computation without losing too much in performance. We have shown that, by increasing the selectivity, we can even gain significantly in precision without losing too much in recall.

One multi-label collection issue we have not considered is inter-class dependency. In some preliminary analysis we found that the correlation among classes is relevant enough to be considered. We could actually benefit from such a correlation to speed up the classification process, by discarding those classes not correlated to the ones we have already found relevant. This relation could probably be used to fight one of the drawbacks found: our recall is very low compared to the precision. If we were able to select those classes that are highly correlated with classes assigned with high precision, we might gain in recall. This will need further investigation.

7 Acknowledgments

This work has been partially supported by Spanish Government (MCYT) with grant TIC2003-07158-C04-04.

References

1. E. L. Allwein, R. E. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. In *Proc. 17th International Conf. on Machine Learning*, pages 9–16. Morgan Kaufmann, San Francisco, CA, 2000.
2. Bruno Pouliquen, Ralf Steinberger, and Camelia Ignat. Automatic Annotation of Multilingual Text Collections with a Conceptual Thesaurus. In A. Todorascu, editor, *Proceedings of the workshop 'Ontologies and Information Extraction' at the EuroLan Summer School 'The Semantic Web and Language Technology'(EUROLAN'2003)*, page 8 pages, Bucharest (Romania), 2003.
3. N. V. Chawla. C4.5 and imbalanced data sets: Investigating the effect of sampling method, probabilistic estimate and decision tree structure. In *Workshop on Learning from Imbalanced Datasets II, ICML*, Washington DC, 2003.
4. C. Drummond and R. C. Holte. C4.5, class imbalance, and cost sensitivity: Why under-sampling beats over-sampling. In *Workshop on Learning from Imbalanced Datasets II, ICML*, Washington DC, 2003.
5. W. Hersh, C. Buckley, T. J. Leone, and D. Hickam. Ohsumed: an interactive retrieval evaluation and new large test collection for research. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 192–201. Springer-Verlag New York, Inc., 1994.
6. N. Japkowicz. Class imbalances: Are we focusing on the right issue? In *Workshop on Learning from Imbalanced Datasets II, ICML*, Washington DC, 2003.
7. N. Japkowicz and S. Stephen. The class imbalance problem: A systematic study. *Intelligent Data Analysis Journal*, 6(5), November 2002.

8. T. Joachims. Text categorization with support vector machines: learning with many relevant features. In C. Nédellec and C. Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, number 1398, pages 137–142, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE.
9. R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proc. of the Fourteenth International Joint Conference on Artificial Intelligence*, pages 1137–1145. Morgan Kaufmann, San Mateo, CA, 1995.
10. D. D. Lewis. Evaluating Text Categorization. In *Proceedings of Speech and Natural Language Workshop*, pages 312–318. Morgan Kaufmann, 1991.
11. M. Martín-Valdivia, M. García-Vega, and L. Ureña López. LVQ for text categorization using multilingual linguistic resource. *Neurocomputing*, 55:665–679, 2003.
12. A. Montejo-Ráez. Towards conceptual indexing using automatic assignment of descriptors. Workshop in Personalization Techniques in Electronic Publishing on the Web: Trends and Perspectives, Málaga, Spain, May 2002.
13. A. Montejo-Ráez and D. Dallman. Experiences in automatic keywording of particle physics literature. *High Energy Physics Libraries Webzine*, (issue 5), November 2001. URL: <http://library.cern.ch/HEPLW/5/papers/3/>.
14. K. Morik, P. Brockhausen, and T. Joachims. Combining statistical learning with a knowledge-based approach - a case study in intensive care monitoring. In *Proc. 16th International Conf. on Machine Learning*, pages 268–277. Morgan Kaufmann, San Francisco, CA, 1999.
15. M. F. Porter. *An algorithm for suffix stripping*, pages 313–316. Morgan Kaufmann Publishers Inc., 1997.
16. B. Raskutti and A. Kowalczyk. Extreme re-balancing for svms: a case study. In *Workshop on Learning from Imbalanced Datasets II, ICML*, Washington DC, 2003.
17. G. Salton, A. Wong, and C. S. Yang. A Vector Space Model for Automatic Indexing. Technical Report TR74-218, Cornell University, Computer Science Department, July 1974.
18. F. Sebastiani. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, 2002.
19. C. J. van Rijsbergen. *Information Retrieval*. London: Butterworths, 1975. <http://www.dcs.gla.ac.uk/Keith/Preface.html>.
20. G. Wu and E. Y. Chang. Class-boundary alignment for imbalanced dataset learning. In *Workshop on Learning from Imbalanced Datasets II, ICML*, Washington DC, 2003.
21. Y. Yang. A study on thresholding strategies for text categorization. In W. B. Croft, D. J. Harper, D. H. Kraft, and J. Zobel, editors, *Proceedings of SIGIR-01, 24th ACM International Conference on Research and Development in Information Retrieval*, pages 137–145, New Orleans, US, 2001. ACM Press, New York, US. Describes RCut, Scut, etc.
22. Y. Yang and X. Liu. A re-examination of text categorization methods. In M. A. Hearst, F. Gey, and R. Tong, editors, *Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval*, pages 42–49, Berkeley, US, 1999. ACM Press, New York, US.