Multi-label Hierarchical Text Classification using the ACM Taxonomy

António Paulo Santos[1], Fátima Rodrigues[1]

[1]GECAD – Knowledge Engineering and Decision Support Group,
Institute of Engineering – Polytechnic of Porto, Portugal
{pgsa, mfc}@isep.ipp.pt

Abstract. Many of the works of text classification involve the attribution of each text a single class label from a predefined set of classes, usually small and flat organized (flat classification). However, there are more complex classification problems in which we can assign to each text more than one class (multi-label classification), that can be organized in a hierarchical structure (hierarchical classification) to support thematic searches by browsing topics of interests. In this paper, a problem of multi-label hierarchical text classification is presented. The experiment involves the creation of a multi-label hierarchical text collection, its pre-processing, followed by the application of different classifiers to the collection, and finally, the evaluation of the classifiers performance.

Keywords: multi-label hierarchical text classification; ACM taxonomy

1 Introduction

The classification of texts consists on the allocation of one or more previously existing categories to text documents, based on their content. More formally, considering a set of categories C = (C_1, C_2,…,$C_{|C|}$) and a set of classified documents D = (d_1, d_2, ..., $d_{|D|}$), using a method or algorithm for learning, the intention is to build a classifier or a classification function which maps the documents into categories. The classifier is then used to classify new documents, not yet rated.

The multi-label hierarchical classification of documents is based on the task of assigning any number of classes, which are organized in a hierarchical structure, to text documents. In the literature there are many contributions about multi-label classification and also many about hierarchical classification. However, if we focus on the combination of these two problems, we find only a few contributions, based in AI techniques, with some limitations. Multi-label classification methods have been categorized into two different groups [16]: *problem transformation methods* and *algorithm adaptation methods*. The methods of the first group are algorithm independent. They transform the multi-label classification task into one or more single-label classification tasks. The methods of the second group extend specific learning algorithms in order to handle directly with multi-label classification. There are multi-label extensions of various algorithms, such as: decision tree [3], support vector machine [4], neural network [23] and k-nearest neighbour method [21].

In single-label classification, the forecast of a document class can only be right or wrong, since the document only belongs to one class. In multi-label classification, the forecast may be right, wrong or partially right, because, in the case documents belong to two or more classes, the projection can hit them all (classes), any of them or just a few of them. Several measures were proposed to evaluate the multi-label classifiers [24]. They are divided into *example-based* and *label-based* evaluation measures [17]. The first are calculated based on the average differences of the actual and projected set of labels over all test examples. The later decomposes the evaluation process into separate evaluations for each label, and then calculates the average of all labels. *Hamming Loss* and *Classification Accuracy* [14] are e*xample-based* evaluation measures. Any known measure for binary evaluation, such as accuracy, precision and recall, can be used as a *label-based* evaluation measure. The calculation of these measures for all labels can be achieved using two averaging operations, called macro-averaging and micro-averaging.

As a set of pre-defined classes we will use the ones defined in the latest ACM Classification Scheme, version 1998[1]. The defined classes are related to computer science and are organized hierarchically into four levels. The first level comprises 11 major partitions subjects codified A...K; these are subdivided into 81 second-level topics, which are further subdivided into third-layer topics. These are then subdivided in uncoded subtopics called subject descriptors. An important aspect to prepare a document for publication by ACM Press is to provide the information to index it according to the ACM classification system. At the moment, the authors, as specialists in the content of their publications, provide the categories and general terms. Instructions and guidelines for this procedure, besides its complexity and extensiveness (about 12 steps), are always dependent on the judgement of each author.

In this article, there are two main contributions: the creation of a truly hierarchical multi-label document collection, extract from the ACM library, and the development of a methodology for multi-label hierarchical text classification composed by various pre-processing techniques and a combination of various classification algorithms.

The paper is organized as follows. In section 2, the creation of the multi-label hierarchical text collection is reported, and also a simplified description of the ACM website extractor/collector system. Section 3 describes the pre-processing techniques used and the representation of texts chosen. Section 4 is about feature selection and the various documents collections created to test the classifiers. The next section is devoted to the creation of the multi-label classifiers. In Section 6, the results of the classifiers´ performance evaluation are presented, and in the last section, conclusions are drawn and some suggestions are made for future work.

[1] Available http://www.acm.org/class/1998/

2 Multi-label Hierarchical Text Collection

The most important element of a classifier is its training set. A training set is just a set of documents that exemplify the different classifications as fully as possible. If the training set is poor, the classifier cannot classify incoming documents correctly.

There are several sets of texts, including Reuters-21578[2], 20 Newsgroups[3], as well as biological data sets ENZYME [20], used and referenced in the literature for multi-label classification. However, these collections are not suitable for the study of multi-label hierarchical classification, because, besides not having an original hierarchical structure (Reuters-21578, Reuters-RCV1), they are not multi-label (20 Newsgroups) or are not even a text collection (ENZYME).

Due to the lack of a truly hierarchical and multi-label collection of texts, we developed a solution able to autonomously navigate through the ACM digital library, in order to collect Web pages and extract the relevant data to build a test collection. From the various types of scientific documents available (journals, magazines, proceedings, among others), we chose to collect the proceedings because this is the largest type of document represented in the ACM library and it covers a large share of the ACM categories involved, thereby exempting the collection of another type of document.

2.1 Architecture

Using the SAX2 API (Simple API for XML version 2) the ACM tree was extracted from a XML file and stored in a database[4]. Due to the large number of Web pages needed and the high volume of data necessary to extract, we designed and implemented a system capable of automatically browsing the relevant Web pages from the ACM website and extract the contents of interest to be stored in the created database. The designed system architecture was based on Google's [2] architecture. Figure 1 shows a simplified architecture of the system, where the Extractor, its key component, is represented. The system works as follows: (1) the Crawler is initially supplied with an address (URL) of the ACM digital library; (2) the Crawler delivers the Web Page to the Extractor; (3) The Extractor consults its set of rules and:

a. If the page has details of a scientific article, it extracts data such as title, keywords, abstract, primary and secondary classification, etc. and save data in the database;
b. If the page points to scientific articles, their links are extracted and saved in the database;
c. If the page is not any of the previous types, then it is not relevant and therefore nothing is done.

(4) The Crawler checks if there are web addresses in the database and if there are, goes back to point 2; otherwise, ends.

[2] http://www.daviddlewis.com/resources/testcollections/reuters21578/

[3] http://people.csail.mit.edu/jrennie/20Newsgroups/

[4] Available at http://www.dei.isep.ipp.pt/~i000313/

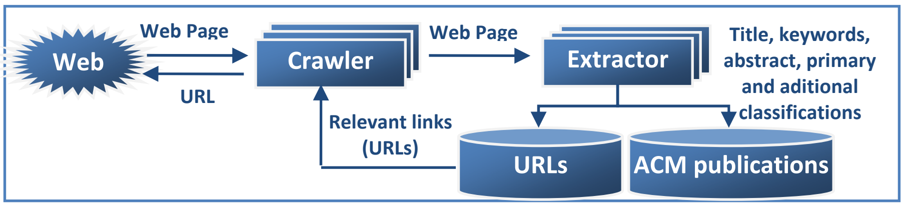

Fig.1. Simplified architecture of the collection and extraction system

The implemented process has the ability to find relevant Web pages and ignore the others. 112,000 Web pages were collected; of those 106,000 correspond to pages with information on scientific documents such as title, abstract, keywords, general terms, ACM classification, authors and the connection (link) to the full document in *pdf* format. The information of each document was obtained first with the identification of relevant Web pages, followed by the acquisition and retrieval of information. The information concerned by each publication was extracted from the web pages based on manually written rules. This was possible because the pages of the ACM site are reasonably structured which facilitates the writing of rules to automatically extract the information. From the documents collected in the ACM portal, it is possible to infer that a document may have between zero and seven primary classifications and between zero and thirty-six additional classifications - such a large number of classifications only occur in extreme cases. Considering the number of classifications, ignoring the distinction between primary and secondary classifications, the collected documents are distributed according to the number of classifications, as shown in figure 2.

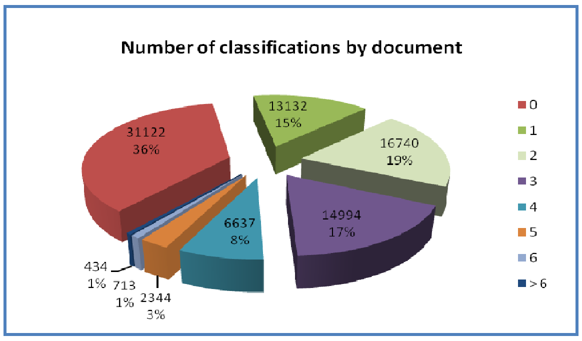

Fig. 2. Classification of documents collected in the ACM Portal

The distribution expresses the following: 31,122 documents do not have any classification, whereas only 13132 documents have one classification and the others have two or more classifications.

3 Pre-processing and Representation of Texts

After obtaining the document text collection, where each training document is composed by title, abstract, keywords and their classifications, it was necessary to preprocess it in order to get better performance from the classification algorithms. In this step, we used the API WVTool 1.1 (Word Vector Tool) [13]. The preprocessing tasks performed were:

- Term normalization, consisting of changing all characters to lowercase, removal of accents and treatment of equivalent terms (although written in different ways - for example, color/colour);
- Segmentation, division of text in single units (procedure known as tokenization), where all the characters (not letters) were recognized as separators, resulting units (tokens) with just letters;
- Stopwords removal, based on a standard English list of words, incorporated into the API;
- Reduction of the words to his radical, by the Stemming process, using the Porter Stemmer algorithm [11];
- Pruning (elimination) in the collection of words with frequency lower than 3. We have not done the elimination of words with frequency greater than a given value, because these words were deleted in the removal of stopwords or excluded in the selection of the most important terms (task reported in the next section).

After the documents pre-processing, each document was represented as a vector of terms to facilitate its manipulation. In this representation, each component of the vector represents a word and has an associated weight according to the TF×IDF (term frequency inverse document frequency) measure. This measure represents the number of times the term occurs in the document, normalized according to the total number of terms in the collection of documents. This measure gives a greater weight to terms that appear in the document more frequently but rarely in the collection of documents. The resulting vector for each document was normalized to Euclidean length.

4 Feature Selection

After the documents pre-processing phase, the number of terms originally in the text collection remained high, although it was considerably reduced. The high number of terms or features is typical of text classification problems. This high number of features is not desirable because it significantly increases the amount of time necessary for a classifier to learn. In fact, not all the terms used in text documents are relevant to describe them (and may even reduce the quality of the classifier´s learning). As a result, it is common to choose a subset made up of the most important terms. To assign a value which represents the importance of a term, there are several measures such as: information gain [12], mutual information, χ^2 (chi-square) and frequency [9]. We apply the measure information gain, since is one of the most effective measures [18].

4.1 Document Collection

As described before, each document in the ACM collection has several primary and secondary classifications (multi-label classification), and each classification is organized in a hierarchical structure of four layers. However, in our experiments we only considered the first two levels of the ACM hierarchy, that is, the documents are classified according to the classes A..K of the first level, and the topics of the second level. From the collection of documents gathered from ACM site, we have created two smaller collections of documents: one with 5000 documents and the other with 10,000 documents. Both are described on table 1.

Table 1. Documents Collection

	Multi-Label 5000	Multi-Label 10000
Number of documents	5000	10000
Total number of labels	11306	23786
Average number of labels per document	2,2612	2,3786
Maximum number of labels per document	14	19
Number of distinct terms after removing *Stopwords*	11743	16475
Number of distinct terms after removing *Stopwords* and pruning of terms with lenght < 3	4467	5697
Average number of documents by category in the 1st level (11 categories)	454,5	909
Average number of documents by category in the second level (81 categories)	61,7	123,4

Note: each document has one or more labels. In other words, a document belongs to one or more categories, and therefore contributes as a unit for one or more categories. Since each of the collections above has a high number of different terms, we created new collections of documents from each of them, each one with the 200, 400, 600, 800 and 1000 most important terms, selected according to the gain of information measure.

5 Multi-label hierarchical classification

5.1 Applied Algorithms

We apllied various plain multi-label classification algorithms: *Binary Relevance* (BR), *Label Powerset* (LP) [16] and *Multi-Label k-Nearest Neighbor* (MLkNN) [21]. The first two are *problem transformation methods*, while the last one is an *adaptation algorithm*. The first two methods were chosen because they correspond to the two most basic approaches to multi-label classification problems. The MLkNN was chosen as representative of the latest methods of *problem adaptation*. The applied methods are in the API Mulan[5] (Multi-label classification). The BR and LP algorithms were applied using the following basic classification algorithms:

– Sequential Minimal Optimization (SMO), to train a classifier based on support vector machines using polinomial kernels [10];
– IBK, which is the implementation of the Weka software of k-Nearest Neighbor Algorithm [1];
– Naive Bayes Multinomial [6];
– SVM (Support Vector Machine), using the library libSVM[6];
– J48, which is the implementation of the C4.5 algorithm [9] for decision trees learning (available in Weka).

With the exception of the SVM algorithm, all other algorithms are in the Weka[7] software. All experiments involving the IBK algorithm were performed with the k-value equal to 1 and 5. All experiments involving the MLkNN were performed with k equal to 1, 5, 10, and 30. The different values assigned to k, both in algorithms IBK and MLkNN, define the number of documents neighbors, for which the algorithm is based to make a decision. In all other algorithms, the parameters used were the default in their APIs.

The three multi-label classification algorithms applied are flat classification algorithms, so their direct application to this problem is not possible; therefore we adopted the local hierarchical approach described in the next section.

5.2 Methods to Handle Hierarchical Problems

5.2.1 Methods of Problem Transformation

A solution to handle the problem of hierarchical classification is to transform the hierarchical structure of categories in a flat structure and thus treat it as a problem of flat classification, for which there are known solutions and with good results [5] [19].

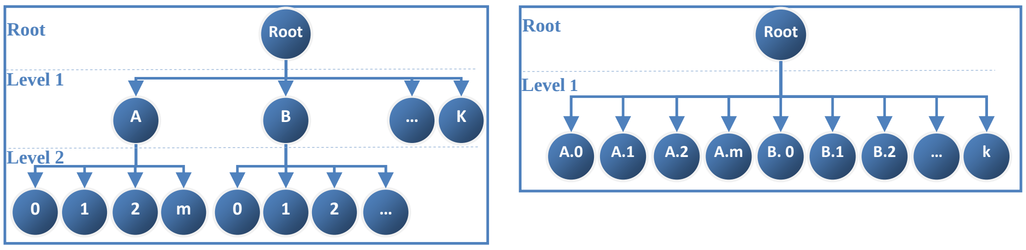

Fig. 3. Transformation of a hierarchical structure into a flat structure

This approach does not make use of the hierarchy of categories, resulting in the loss of this knowledge. Despite the results in [8] were very similar when dealing with the

[5] Available http://mlkd.csd.auth.gr/multilabel.html
[6] Available http://www.csie.ntu.edu.tw/~cjlin/libsvm
[7] Available http://www.cs.waikato.ac.nz/ml/weka

problem of classification in a hierarchical and flat shape, it is our opinion that such results were possible due to the reduced number of categories involved in the problem.

The results reported in [7] with a higher number of categories, support the use of a hierarchy between the categories. For this reason in this work we adopt the local hierarchical approach.

5.2.2 Local Hierarchical Approach

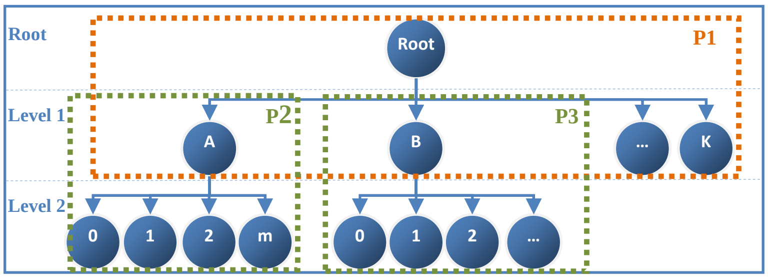

Fig. 4. Local hierarchical approach

In the local hierarchical approach the classification of a new document starts by building one or more classifiers in the root node (P1 problem solving), whose task is to indicate which categories A,B,C,...,K of the first level are relevant to the document. Assuming that A and B were chosen as relevant categories, it is now necessary to go down to the second level and build one or more classifiers, in each of these categories (separately solve the problems P2 and P3). Assuming that the classifier or classifiers responsible for selecting the children categories of node A indicate as relevant category 1, and the classifier or classifiers responsible for selecting the categories of child B indicate as relevant categories 0 and 2, they are then assigned the following ratings to the document: A → 1, B → 0, and B → 2. Note that the details of each category to which the document belongs are made indicating the path from the root, because if a document belongs to a subcategory, it also belongs to all their ancestors.

5.2.3 Global Hierarchical Approach

In the global hierarchical approach (or *big-bang*) [15], the class hierarchy is treated as a whole, and thus only a single classifier is responsible to discriminate all the classes [7]. This approach is similar to the transformation of a hierarchical structure into a flat structure approach, but it somehow considers the hierarchical relationships between classes. For this reason, the use of the flat classification methods in its original form is not possible, it is necessary to do some transformations in order to capture the relationships between classes.

The construction of a classifier using the global approach is more complex than following the local approach: it is computationally more complex and not flexible; for example, each time there is a change in the hierarchy of classes, the classifier needs to be trained again. The local hierarchical approach is identified as computationally more efficient than the global approach, but has a weakness in the spread of errors, that is, a wrong choice of a category in a given level of the hierarchy means that the error is propagated to all its descendants nodes.

6 Evaluation

The best known methods of evaluation of classifiers, such as the holdout method, k-fold cross-validation, leave-one-out and boostrap were designed to assess problems of plain classification. Since this problem is a hierarchical classification problem it was necessary to adapt the method k-fold cross-validation for each level of the hierarchy. As there is still no consensus or a clear trend on the evaluation measures to be applied to multi-label hierarchical classification problems, we chose to implement several measures based on examples.

Note that, for each classifier, all the learning steps (learning, classification and evaluation) were made together and not separately. The experiments were performed following the k-fold cross-validation method with k = 3 (where k is the number of subsets and the number of times that cross-validation process is repeated). Although this value is not as popular as the values k = 10, k = 5, shown to be the most appropriate, given the large number of experiments performed, and the size of text sets.

7 Results

With respect to the ACM tree, we only used the first and second level, because going down to the third level would result in categories with small numbers of documents. In both 5000 and 10,000 documents collections we have applied the various algorithms above refered, using the 200, 400, 600, 800 and 1000 most important terms in each collection, selected according to the information gain measure. Next, we only present the classification results at the second level of the tree, because only at this level the final document classification is obtained, see figures 5 and 6. Among all combinations of algorithms tested we only present the best results that match those obtained with *Binary Relevance* combined with *Naive Bayes Multinomial* (B.R. NB-M.); *Label Powerset* combined with *Sequential Minimal Optimization* (L.P.SMO) and *Multilabel k-Nearest Neighbor* (MLkNN).

The best results were obtained with the algorithms *Binary Relevance* combined with *Naive Bayes Multinomial*. It is clear that the results obtained with this combination of algorithms are independent of both the size of the collection or the number of terms used, that is, the results are similar in both collections, with 5000 and 10,000 documents and are better with smaller number documents terms, that is, 200 terms are sufficient to characterize this type of scientific documents collected from the ACM repository. In what concerns the other algorithms, the algorithm MLkNN gives better results than those obtained with the algorithm L.P.SMO. Regarding the documents collection, the MLkNN algorithm has the best results with the largest collection (10,000 documents) and with the largest number of terms (1000 terms). But the L.P.SMO algorithm has better results with the smallest collection (5000 documents) and also needs a large number of terms (1000).

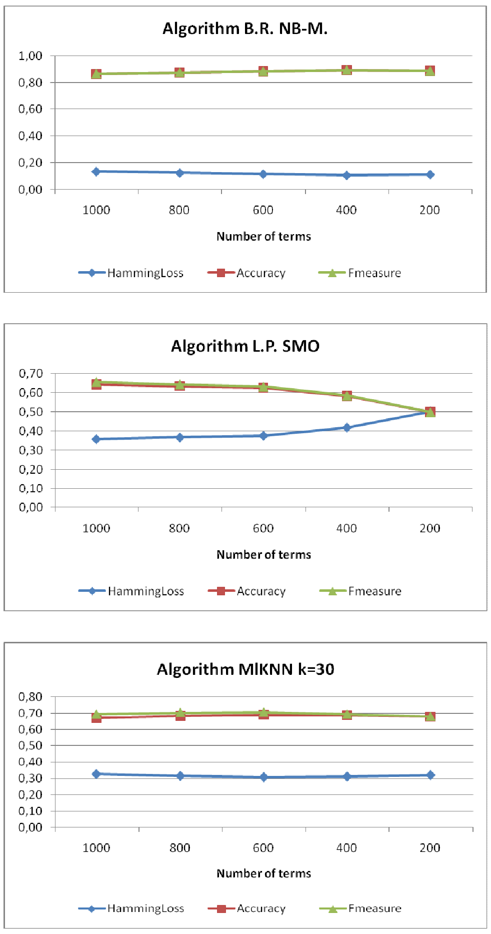

Fig. 5. Results of the different algorithms on dataset 5000, according to the number of terms

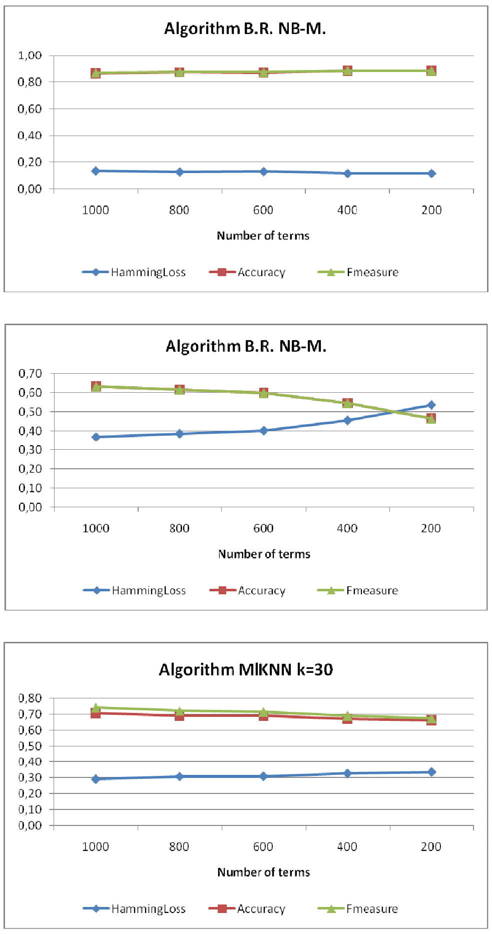

Fig. 6. Results of the different algorithms on dataset 10000, according to the number of terms

8 Conclusions and Future Work

Text classification has received the attention of researchers, since the mid 90´s. However, the multi-label hierarchical text classification still remains actual and offers exciting challenges, which give space for new research, and optimization of contributions already available. In this paper, we give an overview of hierarchical classification problems and their solutions. A multi-label hierarchical text collection classified according to the ACM scheme was created and pre-processed using various techniques. We have tested two problem transformation methods, *Binary Relevance* and

Label Powerset, combined with various classification algorithms and also an adaptation method, *Multilabel k-Nearest Neighbor*. All of them were evaluated using two text collections with different number of terms. The best results were obtained with the combination *Binary Relevance* with *Naive Bayes Multinomial*. From the various experiences performed we can also conclude that this combination does not depend of both the size of the collection and the number of different terms used. We want to stress that these conclusions are only valid for the hierarchical multi-label document collection extracted from the ACM library.

In addition to the work done in this article, here are some proposals for future work:

– Investigate ways to recover from errors in a given level, that is, prevent the spread of errors to lower levels of the hierarchy, which is the main drawback of hierarchical local approach;

– Investigate measures for the evaluation of multi-label hierarchical classification that takes into account the distance in the tree provided between the predict class and the correct class;

– Conducting experiments using a hierarchical approach that combines at each level different classifiers using boosting, bagging or stacking.

Acknowledgments. We thank the Foundation for Science and Technology for the support expressed by funding the research project COPSRO - Computational Approach to Ontology Profiling of Scientific Research Organizations. The authors also thank all the support provided by GECAD.

References

1. Aha, D., Kibler D., Albert M.: Instance-based learning algorithms. *Machine Learning 6*, pp. 37--66 (1991)
2. Brin, S., Page, L.: The Anatomy of a Large-Scale Hypertextual Web Search Engine. In Proceedings of the seventh international conference on World Wide Web 7 Computer Networks, pp. 107--117 (1998)
3. Clare, A., King, R.: Knowledge discovery in multi-label phenotype data. In Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery, pp. 42--53. Freiburg: Germany (2001)
4. Godbole, S., Sarawagi, S.: Discriminative methods for multi-labeled classification. In Proceedings of the 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 22--30 (2004)
5. Joachims, T.:Text categorization with support vector machines: learning with many relevant features. In Proceedings of ECML-98, 10th European Conference on Machine Learning pp.137--142 (1998)
6. John, George H., Langley P. Estimating continuous distributions in bayesian classifiers. In *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence, pp.* 338–345 (1995)
7. Kiritchenko, Svetlana. “Hierarchical Text Categorization and Its Application to Bioinformatics.” Ph.D., Faculty of Engineering University of Ottawa, Canada, (2005)

8. Koller, D., Sahami, M.: Hierarchically classifying documents using very few words. In Proceedings of 14th International Conference on Machine Learning ICML-97, pp.170--178 (1997)

9. Mladenic, D., Grobelnik, M.: Feature selection for unbalanced class distribution and Naive Bayes. In Proceedings of the 16th International Conference on Machine Learning, San Francisco, CA: Morgan Kaufmann Publishers, pp.258--267 (1999)

10. Platt, J.: Fast training of support vector machines using sequential minimal optimization. *Scholkopf, B., Burges, C., Smola, A, eds: Advances in Kernel Methods - Support Vector Learning.* MIT Press (1998)

11. Porter, F.: An algorithm for suffix stripping. Program, 14 no. 3, pp. 130--137 (1980)

12. Quinlan, R.: Constructing Decision Tree in C4.5: Programs for Machine Learning. *Morgan Kaufman Publisher*,.pp. 17--26 (1993)

13. Rousu, J., Saunders, C., Szedmak, S., Shawe-Taylo, J.: Kernel-based learning of hierarchical multilabel classification models. In Journal of Machine Learning Research, (7) pp. 1601--1626 (2006)

14. Schapire, E., Yoram ,S.: BoosTexter: A boosting-based system for text categorization. Machine Learning (39) pp.135--168 (2000)

15. Sun, A., Ee-Peng, L.: Hierarchical Text Classification and Evaluation. In Proceedings of IEEE International Conference on Data Mining (ICDM 2001), pp. 521--528. California, USA (2001)

16. Tsoumakas, G., Katakis, I.: "Multi-Label Classification: An Overview." International Journal of Data Warehousing and Mining 3(3):1--13 (2007)

17. Tsoumakas, G., Vlahavas I.: Random k-Labelsets: An Ensemble Method for Multilabel Classification. In *Proceedings of the 18th European Conference on Machine Learning*, pp. 406--417, Springer Verlag, LNAI 4701, Warsaw, Poland (2007)

18. Yang Y. and J. Pedersen: A comparative study on feature selection in text categorization. In J. D. H. Fisher, editor, The Fourteenth International Conference on Machine Learning (ICML'97), pp. 412-420. Morgan Kaufmann (1997)

19. Yang, Y.: An evaluation of statistical approaches to text categorization. Journal of Information Retrieval Vol.1, No.1/2 (1999)

20. Wurst, M.: The Word Vector Tool.User Guide/Operator Reference/Developer Tutorial (2007)

21. Zhang, L., Zhou., Z.: A k-nearest neighbor based algorithm for multi-label classifcation. In Proceedings of the 1st IEEE International Conference on Granular Computing, pp. 718--721 (2005)

22. Zhang, L.,Zhou Z.: Ml-knn: A lazy learning approach to multi-label learning. *Pattern Recognition,* 40(7), pp. 2038--2048 (2007)

23. Zhang, M., Zhou., Z.: Multi-label neural networks with applications to functional genomics and text categorization. IEEE Transactions on Knowledge and Data Engineering 18, pp. 1338--1351 (2006)

24. Zheng, Z., Wu, X., Srihari, R.: Feature selection for text categorization on imbalanced data. SIGKDD Explorations Newsletter, 6(1), pp. 80--89 (2004)