Automatic Web Page Classification in a Dynamic and Hierarchical Way

XIAOGANG PENG & BEN CHOI
Computer Science, College of Engineering and Science
Louisiana Tech University, Ruston, LA 71272, USA
pro@BenChoi.org

Abstract

Automatic classification of web pages is an effective way to deal with the difficulty of retrieving information from the Internet. Although there are many automatic classification algorithms and systems that have been proposed, most of them ignore the conflict between the fixed number of categories and the growing number of web pages going into the system. They also require searching through all existing categories to make any classification. We propose a dynamic and hierarchical classification system that is capable of adding new categories as required, organizing the web pages into a tree structure, and classifying web pages by searching through only one path of the tree structure. Our test results show that our proposed single-path search technique reduces the search complexity and increases the accuracy by 6% comparing to related algorithms. Our dynamic-category expansion technique also achieves satisfying results on adding new categories into our system as required.

1. Introduction

The World Wide Web is growing at a great speed but the documents in the Web do not form a logical organization and inevitably making the manipulation and retrieval difficult. The need for mechanisms to assist in locating relevant information becomes more and more urgent. One of the solutions to assist in retrieving documents on the Web is provided by classified directories [5]. However, current systems, such as Yahoo [30] still require human labor in doing the classification. Whether manual classification is able to keep up with the growth of the Web remains a question. First, manual classification is slow and costly since it relies on skilled manpower. Second, the consistency of categorization is hard to maintain since different human experiences are involved. Finally, the task of defining the categories is difficult and subjective since new categories emerge continuously from many domains. Considering all these problems, the need of automatic classification becomes more and more important.

In this paper we present an automatic web page classification algorithm. The algorithm, unlike others, stresses the dynamic growing issue. It considers the hierarchical structure information for improving the classification accuracy. The core of the algorithm is a hierarchical classification technique that assigns a web page to a category. The number of web pages on the Web increases continuously in great speed and thus it is impossible for a fixed category set to provide accurate classification. To address this problem, we propose and implement a dynamic expanding technique.

1.1 Related Research

This paper relates to text learning and document classification. Text learning is a machine learning method on textual data that combines information retrieval techniques and is used as a tool to extract the content of textual data. A simple, yet limited, document representation (DR) is the "bag-of-words" text DR [12] [14]. Many experiments have been done to improve the performance of the DR. For example, Mladenic [18] extends the "bag-of-words" to the "bag-of-phrases" representation. Chan [4] also suggested that using phases is a better choice than using single words. The goal of using phrases as features is to attempt to preserve the information left out by the "bag of words" methods. A document representation called "feature vector representation" uses a feature vector to capture the characteristics of the document by an "N-gram" feature

This research was supported in part by Center for Entrepreneurship and Information Technology (CEnIT), Louisiana Tech University, Grant iCSe 200123.

selection. An N-gram feature could be a word or a sequence of N words. A vector consists of a feature along with the occurrences of that feature within the document. Experiments show that N ranging from two to three is sufficient in most classification systems.

In information retrieval, TFIDF (Term Frequency–Inverse Document Frequency) classification algorithm is well studied [25]. Based on the document vector model, the distance between vectors is calculated by the cosine of the angle between them for the purpose of classification. Joachims [11] analyzed the TFIDF classifier in a probabilistic way based on the implicit assumption that the TFIDF classifier is as explicit as the Naïve Bayes classifier. By combining the probabilistic technique from statistic pattern recognition into the simple TFIDF classifier, Joachims proposed the PrTFIDF classifier with the formula:

$$P(d \mid c_j) = \sum_{w \in (d \cap c_j)} \frac{P(w \mid c_j) * P(c_j)}{\sum_{c \in C} P(w \mid c) * P(c)} * P(w \mid d) \tag{1.1}$$

Where c and c_j are categories taken from a category set C, $P(d|c_j)$ is the probability for a document d given a category c_j, and $P(w|d)$ is the probability of a feature w given the document d.

The PrTFIDF classifier optimizes the parameter selection in TFIDF and reduces the error rate in five out of six reported experiments by 40%. Other more sophisticated classification algorithms and models were proposed including: multivariate regression models [8][26], nearest neighbor classifiers [31], Bayesian classifiers [15], decision tree [15], Support Vector Machines [7][10], voted classification [28]. Tree structures appear with all of these systems. Some proposed systems focus on classification algorithms to improve the accuracy of assigning testing documents to related catalogs [11], while others go even further by taking the classifier structure into account [12].

1.1 Organization

This paper is organized into the following sections: Section 2 describes our hierarchical classification module. Section 3 describes our dynamic expansion module. In Section 4 we use Yahoo structure as a test base and conduct several experiments to evaluate our system. Finally, in Section 5 we draw conclusions and provide future research.

2. Our Proposed Hierarchical Classification

In this section, we propose a new classification algorithm on a hierarchical structure. We first provide an overview of the algorithm then detail follows. The algorithm consists following stages: (1) generating category information tree, (2) hierarchical feature propagation, (3) feature selection on category information, and (4) single path traversal.

2.1 Generating Category Information Tree

For a web page classification system, the first step is to define the concept hierarchy using domain knowledge and to collect text data that corresponds to the concept hierarchy. The data is collected into an appropriate format for classification by a text-learning algorithm. The characteristics of each category are represented as a "bag of words" or a feature list based on the "well-grained text-learning method" [24]. Categories are arranged in a hierarchical tree structure. Each tree node represents one category. A child note of any given node represents a subcategory under the given node.

2.2 Hierarchical Feature Propagation

Many existing category structures are unbalanced hierarchical structures. In order to ensure the actual containment relation, the feature information of a category is propagated upward from leaf nodes to the root node of our classification tree. For example, Mladenic [22] studied Yahoo unbalanced structure and proposed an algorithm for featuring propagation. The algorithm takes care of the structure of the tree hierarchy. As proposed in [22], with a tree T rooted at node N having k sub-trees (SubTi $_{(i=1\ldots k)}$), we can calculate the probability for a feature w belonging to a category after propagation as follows:

$$P(w \mid T) = \sum_{i=1}^{k} P(w \mid SubTi) * P(SubTi \mid T) + P(w \mid N) * P(N \mid T) \tag{2.1}$$

Where $P(w|T)$ is the propagated feature probability given tree T and $P(w|N)$ is the probability of the feature w in the node N before the propagation, which can be calculated by dividing the particular term frequency by the total term frequency. $P(SubT_i|T)$ and $P(N|T)$ are the weight factor of the sub-tree $SubTi$ given tree T and the probability of current node given tree T.

In the generated classification tree (from stage 1), each node represents a category by a feature list. The propagated feature list is generated by adding the original feature list of the current node and all the propagated feature lists of the sub-categories and by assigning different weights. By propagating in this way, the feature list captures the characteristics of a sub-tree rooted in the current node rather than in an isolated category set.

2.3 Feature Selection on Category Information

It is clear that what really contributed in distinguish between categories are those unique features belonging to the categories. Because of the feature propagation, these unique features will be weighted and propagated upwards and become the features of the parent category. In this case, by tracing these unique features it is easy to locate the correct category. Due to the uniqueness of the features, there is only one path in the tree for reaching the features. This phenomenon provides a foundation for our single path classification algorithm.

Mladenic and Grobelnic [20] conducted similar experiments in feature selection on Yahoo category tree and proposed Odd Ratio measurement, but they used the complement of a note from the entire tree as negative examples to calculate the Odd Ratios.

The goal of our feature selection is to compare the features in a node to its sibling nodes and try to distinguish the unique features. In order to do this, we take advantage of the feature propagation and use the features of the parent node as negative examples to determine a ranking. After propagation, each propagated feature probability is actually the weighted sum of the same feature probability from the sub-trees and the current node itself. We proposed the following formula for determining the uniqueness ranking $R_c(w)$ of a feature w of a category c.

$$R_c(w) = \frac{P(w \mid c) * P(SubT_C \mid T)}{P(w \mid parent)} \tag{2.3}$$

Where c is the current node and *parent* is the parent node of c. $P(SubT_c|T)$ is the weight factor that assigned to node c when it is propagated to the parent. If a feature is unique in one node, it is the only source that can propagate to the parent feature list. In this case, we get the maximum value of the formula, which is 1. The unique features will be considered as key features that differentiate the node from its siblings and forms the basis for our single path traversal algorithm.

2.4 Single Path Traversal

The concept of text classification requires the use of a classifier to assign values to a document and to each catalog. Matching a document feature values to a category feature values, the catalog with a global maximum value is considered the correct place to hold the document. Most automatic classification researches concentrated on the global search algorithm. They treat all categories in a flat structure when trying to find the maximum. It follows that in order to find the category with the greatest value, it is necessary to compare all the categories.

In the tree structure that we have configured, we claim that traveling one path is sufficient to achieve this goal. If there are N categories in the tree, the complexity of searching all categories is $\theta(N)$, but by our single path algorithm it is merely $\theta(\log(N))$.

The idea of the single path traversal is to eliminate the impact of other branch in the tree. After feature propagation, the propagated feature list of the parent node is a scaled summation of the propagated feature lists of its children. Thus, by checking the parent node we can know the information of its descendents. In our tree structure, all the features are propagated upwards with the ranking function of formula 2.3, so each category has unique features of its own to differentiate from its siblings. These two steps make the single path traversal possible.

The first step of the single path traversal is to discriminate sibling nodes in each level and find a correct path for the incoming web page. In order to determine the discriminating probability for each node, we only consider the nodes at each level and apply the PrTFIDF formula 1.1 on the features with a ranking of 1, which indicate a unique feature. At each level, we chose the node with the maximum discriminating probability as the starting point for the next iteration. Recursively applying this rule creates a path from the root of the tree to one of the leaf nodes.

Then following this path we apply the PrTFIDF classifier again using all the features of the nodes belonging to the path, to get the actual probability for the page with categories within this path. By picking the node along the classification path with maximum actual probability value, we determine the candidate category for the page.

3. Dynamic Expansion and Updating

In this section, we describe our proposed new dynamic tree expansion algorithm. As more and more documents being put into a catalog set, the diversities of the documents make the existing catalogs unable to guarantee classification accuracies. The problem of how to dynamically generate more sub-catalogs for the existing catalog set becomes evident. The highlight of our approach is as follow; details are provided in following subsections. Based on statistical results, we determine a set of criteria and check whether the criteria have been met for putting a page under the current category. If the criteria have not been met, we will create a new node for the page. As the results, the tree will be expanded by adding a new category. An updating algorithm is also introduced to incorporate the expansion information into the existing catalog set.

3.1 Dynamic Expansion

As the number of pages that need to be classified grows larger and because of their diversity, the original number of categories does not always fit the true content of the incoming pages. It is necessary for expending the category tree to accommodate all the pages in order to yield accurate results. The criteria for creating new categories must be established. There are two types of expansion, deeper and wider, and two thresholds will be used to determine the criteria.

The two thresholds B and D are obtained in the following ways. We take a set of sample pages from a category and calculate the probability of these pages relating to its category. We denote the resulting values for each category as S_i for i from 1 to the number of all categories n. Then, we compute the normal distribution of all sample S_i for i = 1 to n, and obtain the mean μ and the standard deviation *d*. We set $B = \mu - d$ and $D = 2d$.

We define the relation between a document and the category it should be placed as the "belongs to" relation. It is a basic assumption that the probability of a page given a category having the "belongs to" relation will have a maximum value. In expansion, we set a lower bound for this relation, threshold B will be used to ensure the maximum characteristic of the relation.

A deeper expansion happens when the maximum actual probability value (obtained from single-path search described in the last subsection) is smaller than threshold B. That is although the value is the maximum found, its corresponding category is not considered to be sufficiently suitable for the new page. A new sub-category under the category is then created to hold the new page.

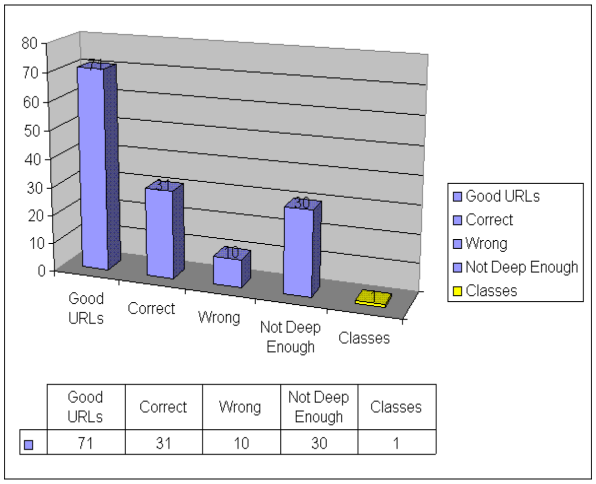
A wider expansion occurs in the following situation: the probability of the page comparing to that the candidate category (the one with the maximum actual probability value) is much bigger than the probability of the page and that the candidate's children categories. The difference of probability represents the distinction between the page and the candidate's children nodes. Ignoring this distinction will cause the inconsistency of the "belongs to" relation between parent and children. When updating the category information after a page is put into a category, those features, which contribute to the difference, will also be incorporated into the category. This makes the relation between the category and the existing children more and more distinct. By creating the new category, we put the distinction to the siblings other than making the relation between parent and children far apart. Thus, the newly added features will only heavily affect the new created category. When propagating the new features to the candidate category, the weight factor is used to reduce the effect of the new feature. This reduces the inconsistent effect.

Based on these two cases, the threshold B ("belongs to" threshold) and the threshold D (difference threshold) are used as criteria to determine when expansion is needed. For deeper expansion, the probability for the new page (document *d*) given the candidate node is less than B, that is $P(d|c) < B$. In this case, we will create a new category. For wider expansion, the candidate note is not a leaf node of the tree. We need to check the probability of the page given each sub-categories of the candidate category $P(d|Sub_i)$. If the difference of $P(d|c)$ and the maximum number of the $P(d|Sub_i)$ is out of the range of threshold D, that is $P(d|c) - Max(P(d|Sub_i)) > D$, then the new page is considered to be substantially different from any of its sub-categories. In this case, we will create a new category.

3.2 Updating Category Information

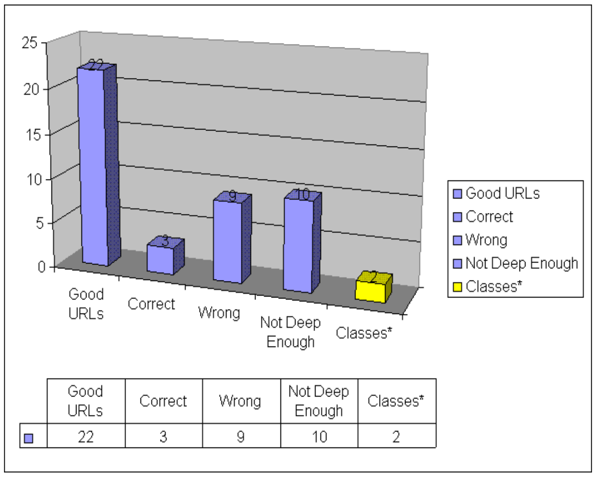
When a page is assigned to a category, whether a page is just created or an existing one, the feature vector of the page will be contributed to the category information. It is necessary to update the category information feature list to maintain the concordance and the hierarchical structure and expansion of the vocabulary will inevitably change the characteristic of some of the categories.

After the page has been put into a category, the page is considered to be one part of the category, so it will contribute its own characteristics to the category. Both the page and category information are represented as feature vectors. Merging the page vectors into the category vectors is a solution for updating the category information. Since the page features are selected by a "well-grained text-learning method" [24] with those low frequency patterns removed, we assume that the extracted



	Good URLs	Correct	Wrong	Not Deep Enough	Classes
🟦	71	31	10	30	1

Figure 4.1 Results for accuracy of deeper expansion



	Good URLs	Correct	Wrong	Not Deep Enough	Classes*
🟦	22	3	9	10	2

Figure 4.2 Experiment results of wider expansion

features are considered to be relevant to represent the page. We use the following formula for updating:

$$P'(w \mid N) = \frac{\mid V \mid * P(w \mid N) + \mid Vpage \mid * P(w \mid Page)}{\mid V \mid + \mid Vpage \mid} \tag{3.1}$$

Where |V| represents the size of the category vector and the |Vpage| is the page vector size.

After merging page vectors, the category information is changed, so the ranking and propagated feature probability should be recalculated. Since those changes happen just along the classification path, the updating takes place only along the single path.

.

4. Experiments and Results

We design experiments to test two aspects of our system: accuracy in dynamic expansion and performances of the single path traversal. The root of our experiment is set to Yahoo /Science/Engineering/, one of the sub-categories of Yahoo's classification tree. We also chose the categories that are strictly following the levels of this root category; that is, we eliminated those categories that either go to another root category or do not follow the level structure. As we have noticed, many of the outgoing URLs in Yahoo are not accessible. We chose the Science/Engineering as the root because it is newly generated and covers 4068 outgoing URLs as advertised. Our expectation is that it will somewhat reduce the number of outdated outgoing URLs and provide enough testing examples for our system.

4.1 Machine Learning Setting

Considering processing time for testing purposes, we direct our program in getting the category information of three levels in the Yahoo "Science/Engineering" sub-tree. Labrou and Finin [13] compared several different ways in describing the category information and the web page information. By their experiments, they pointed out that, the best way of describing category was entry summaries and entry titles; correspondingly the best choice of web page description (called entry in their paper) was the entry summaries. Because of this, when generating the category information, we use summaries that are generated by man power and already there in Yahoo website, instead of using the actually website contents to generate the category information tree.

The testing examples of our system are actual website contents whose URLs are taken from three levels of the Yahoo sub-tree rooted in "Science/ Engineering." Some non-text format paged associated with the URLs cannot yet be classified, for example, jpeg, swf, and gif files. We only use the URLs whose pages having more than 70 features after our well grain 3-gram feature extraction, and these URLs are called "good" URLs.

The two global variables (thresholds) are generated at the machine-learning step using the statistic of the page-category probability. All the categories at the second and the third levels of "Science/ Engineering", except the "Science/ Engineering/organizations" category, are defined as existing categories. One third of the URLs that belong to those categories are taken to calculate the two thresholds based on the ways that we have described in section 3.1.

Table 4.1 Results of comparing two algorithms

Same classification results		
Correct results	Wrong results	
333	6	
Different classification results		
Correct results by Single path algorithm	Correct results by thorough search algorithm	Wrong results by both algorithm
58	30	37

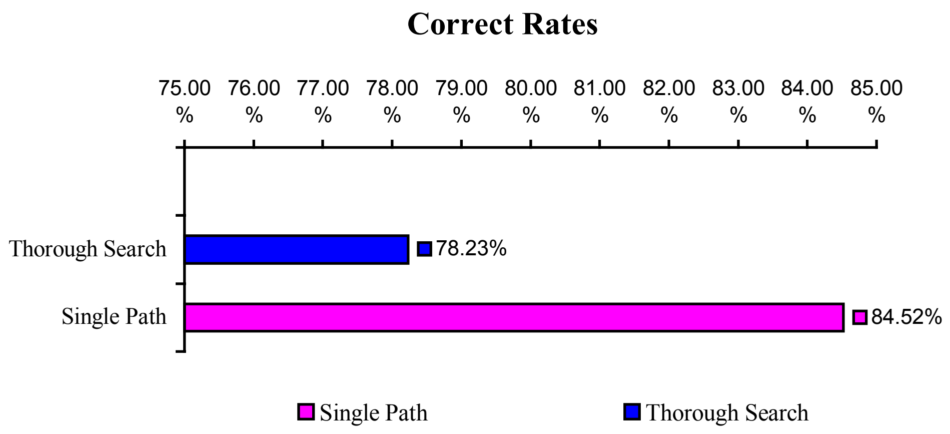


Figure 4.3 Correct rates of both algorithms

4.2 Expansion Tests

Our expansion experiments are designed to test two kinds of accuracies: deeper test – testing the accuracy for deeper expansions, (see results in Figure 4.1), and wider test – testing the accuracy for wider expansions, (see results in Figure 4.2).

All of the URLs that have the "belong to" relation to the category "Science/Engineering/organization" are considered to be a "new" group and used to test the wider expansion. In this experiment, the number in "correct" column means how many pages are classified to an expanded category that is under the selected "Science/ Engineering/organization". If the page is classified to a category that contains the category where this page comes from, it is called "Not Deep Enough." The column "Classes" keeps the number of the newly expanded category. Since all the URLs are taken from a same category, we are expecting the number to be 1. In deeper test, the testing URLs come from "Science/ Engineering/ Civil _Engineering /Institutes/." Similar for wider test, we have same settings for the experiment results. Experiment results are provided on figure 4.1 and 4.2.

4.3 Testing our Single Path Algorithm

In order to test the accuracy of single path algorithm, we compare the actual classification results of the single path algorithm to the one that searches all categories. Using 33% of all the web pages in Yahoo "Science/Engineering" tree as testing web pages, we compare the two algorithms by the accuracy and the effectiveness, and the result is shown in Table 4.1 and Figure 4.3.

By the results, we can see that the two algorithms have the same result is more than 73% of all testing cases. Surprisingly, even when we have ignored most of the branches of the tree, the single path classification still has an accuracy of 84.26%, which even outperforms by more than 6% the accuracy that was achieved by the thorough search algorithm. From these results, we can see the advantage of our single-path search algorithms.

5. Conclusion

In this paper, we describe an approach that utilizes class hierarchies for improving text classification. Our single path classification algorithm, in the hierarchical classification module, reduces the computational expense compared to the thorough search algorithm that is used by most of the existing classification algorithms. By distinguishing the siblings, the algorithm recognizes a correct path containing the destination category. The algorithm is successful not only in saving computational resources but even improving the correct hits to a higher percentage. Our experiment also shows that because of the diversity of contents of the pages, the thorough search algorithm sometimes cannot tell the key information from the common information since it treats all the information equally. The single path algorithm avoids this problem by only considering unique features in discriminating siblings. Our experiments support that the single path algorithm is more competent both in time and in accuracy.

The expansion and updating modules emphasize the developing of a dynamic system. The expansion algorithm creates a new category when it detects that the page is not similar enough for the current node or might affect the containment relation between current node and its children. The updating algorithm keeps the tree database in a valid state. With expansion and updating, our database grows more diverse in order to proficiently categorize more incoming web pages.

Finally, since the Internet is growing in great speed but the arrangement of the web pages do not have a logical or semantic organization, to find a structured semantic Internet, we should find a standard organization for all Internet data. Since Internet resources can be considered as different kinds of information units, for grouping proposes, classification is perhaps the most appropriated way to organize them. Our system provides a starting point. Hierarchical structure and dynamic growing mechanism can be considered as bases of a structured Internet. We expect that in the near future, when all Internet resources can be classified, the whole Internet will be converted to a well-structured system based on certain classification standard. To achieve this goal much future research remains to be done.

References

[1] AltaVista, http://altavista.digital.com

[2] A. Berker and V. Mittal, “OCELOT: a system for summarizing web pages”, In Proceedings of SIGIR, 144-151, 2000

[3] M. CatePazzani and D. Billsus, “Learning and Revising User Profiles: The Identification of Interesting Web Sites”, Machine Learning 27, 313-331, 1997

[4] Philip K. Chan, “A non-invasive learning approach to building web user profiles”, KDD-99 Workshop on Web Usage Analysis and User Profiling, 1999

[5] Chandra Chekuri, Michael H. Goldwasser, Prabhakar Raghavan and Eli Upfal, "WebSearch Using Automatic Classification", In Proceedings of the Sixth International World Wide Web Conference, 1997

[6] P. Dominggos and M. Pazzani, “On the optimality of the simple Baysian classifier under zero-one loss “, Machine learning 29, 103-130, 1997

[7] Susan Dumais, Hao Chen, “Hierarchical Classification of Web Content”. Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval, 2000

[8] N. Fuhr, S. Hartmanna, G. Lustig, M. Schwantner, and K. Tzeras, “A rule-based multi-stage indexing system for large subject fields”, Proceedings of RIAO'91, 06-623, 1991

[9] HotBot, http://www.hotbot.com

[10] Thorsten Joachims, “Text categorization with support vector machines: Learning with many relevant features”, Proc. 10th European Conference on Machine Learning (ECML), Springer Verlag, 1998

[11] Thorsten Joachims. “A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization”, In International Conference on Machine Learning (ICML), 1997

[12] D. Koller and M. Sahami, “Hierarchically classifying documents using very few words”, Proceedings of the 14th international Conference on Machine Learning ECML98, 1998

[13] Yannis Labrou and Tim Finin, “Yahoo! as an ontology – using Yahoo! Categories to Describe Document” In CIKM '99. Proceedings of the Eighth International Conference on Knowledge and Information Management, 180-187, ACM, 1999

[14] K. Lang, “Newsweeder: Learning to filter news”, In Proceedings of the 12th International Conference on Machine Learning, 331--339, 1995

[15] D.D. Lewis, and M. Ringuette, “A comparison of two learning algorithms for text categorization”, Third Annual Symposium on Document Analysis and Information Retrieval (SDAIR'94), 81-93, 1994

[16] Lycos, http://www.lycos.com

[17] Daniel Marcu. "From Discourse Structures to Text Summaries", Proceedings of the ACL/EACL-97 Workshop on Intelligent Scalable Text Summarization, 1997

[18] Dunja Mladenic and Marko Grobelnik, "Word sequences as features in text-learning", In Proceedings of ERK-98, the Seventh Electro-technical and Computer Science Conference, 145--148, 1998

[19] Dunja Mladenic, "Feature subset selection in text-learning", In Proceedings of the 10th European Conference on Machine Learning ECML98, 1998

[20] Dunja Mladenic and Marko Grobelnik, "Feature selection for classification based on text hierarchy", Working Notes of Learning from Text and the Web, Conference on Automated Learning and Discovery, 1998

[21] Dunja Mladenic, "Turning Yahoo! into an Automatic Web-Page Classifier", Proceedings of the 13th European Conference on Artificial Intelligence ECAI'98, 473- 474, 1998

[22] Dunja Mladenic, "Machine Learning on non-homogeneous, distributed text data", PhD thesis, University of Ljubljana, Slovenia, 1998

[23] C. D. Paice, "Constructing Literature Abstracts by Computer: Techniques and Prospects", In Information Processing & Management, 26(1), 171--186, 1990.

[24] Xiaogang Peng, "Automatic Web Page Classification in a Dynamic and Hierarchical Way", MS Thesis, Louisiana Tech University, 2002

[25] G. Salton, and C, Buckley, "Term Weighting Approaches in Automatic Text Retrieval", Technical Report, COR-87-881, Department of Computer Science, Cornell University,November

[26] H. Schutze, D. Hull, and O.J. Pedersen, "A comparison of classifiers and document representations for the routing problem", Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 229-237, 1995

[27] C.J. van Rijbbergen, D.J. Harper, and M.F. Porter, "The selection of good search terms", Information Processing & Management, 17, 77-91, 1981

[28] S.M. Weiss, C. Apte, F. Damerau, D.E. Johnson, F.J. Oles, T. Goets, and T. Hampp, "Maximizing text-mining performance", IEEE Intelligent Systems, 14(4), 63--69, 1999

[29] Michael J. Witbrock and Vibhu O. Mittal, "Ultra-Summarization: A Statistical Approach to Generating Highly Condensed Non-Extractive Summaries", 1999

[30] Yahoo! http://www.Yahoo.com

[31] Y. Yang and O.J. Pedersen, "A comparative Study o Feature Selection in Text Categorization", Proc. of the fifth International Conference on Machine Learning ICML97, 412-420, 1997