A new Effective Approach for Categorizing Web Documents

Claus-Peter Klas, Norbert Fuhr
University of Dortmund
{klas,fuhr}@ls6.cs.uni-dortmund.de

Abstract

Categorization of Web documents poses a new challenge for automatic classification methods. In this paper, we present the megadocument approach for categorization. For each category, all corresponding document texts from the training sample are concatenated to a megadocument, which is indexed using standard methods. In order to classify a new document, the most similar megadocument determines the category to be assigned. Our evaluations show that for Web collections, the megadocument method clearly outperformes other classification methods. In contrast, for the Reuters collection, we only achieve mediocre results. Thus, our method seems to be well suited for heterogeneous document collections.

1 Introduction

There are two methods for finding information on the WWW. One can either use a search engine (like e.g. Altavista, Google and MetaCrawler) or one can browse through a Web catalog (like e.g. Yahoo). However, in order to enter a page into a Web catalog, a human has to view the page first and assign it to a category. If automatic categorization is applied here, this process could be speeded up and be performed with less intellectual effort.

However, classification of Web documents is different from categorizing classical collections (like e.g. Reuters or OHSUMED). Whereas documents in the latter collections all have a similar structure and are about the same length, Web documents are rather heterogeneous and can contain anything from advertisement banners to programming code. They have a richer structure and their length varies to a great extent. Standard classification algorithms perform very well on the classical collections, but have problems when being applied to Web documents. They have to be modified in order to achieve good results for Web collections.

There are several techniques for automatic classification (see e.g. the survey in [Yang 99]). Common to all approaches is the following procedure: A collection is divided into a test and a learning sample. The learning sample is used for building representations of each category, e.g. a single vector as in the Rocchio method or as the set of all documents from this category as in kNN. Then each document from the test sample has to be classified using the representations constructed before.

In this paper we propose a new, simple and effective approach for categorizing Web documents.

We evaluate our approach using three different collections. The first is the "Computers and Internet" subtree of the Yahoo[1] collection. The second is the complete set of the German Dino-Online catalog[2]. In order to compare the new approach with other classification methods, we use the well known Reuters-21587[3] collection.

In the following, we first describe our new approach. Then we present the evaluation on the three different collections. Finally, we conclude with proposals for future work.

2 Classification with megadocuments

Our approach is a variation of the idea of using category-specific centroid vectors. For comparison, the basic idea of the Rocchio method is the following: Based on the vector model, the centroid of all document vectors (of documents belonging to the same category) is used as a representative of this category. When being applied to Web documents, this method suffers from the fact that documents are rather heterogeneous; thus averaging over a rather heterogeneous set of document vectors may not lead to good performance (see e.g. the results from Chakrabarti et al. below).

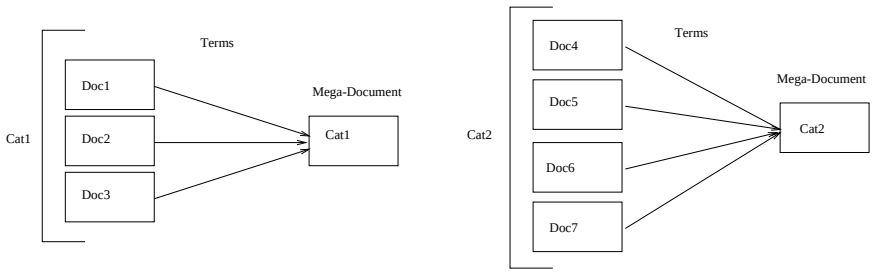

Figure 1: Document generation

In order to cope with heterogeneity, we take a different approach (figure 1): We first concatenate the texts of all documents belonging to a category, thus, achieving a so-called megadocument. In the next step, we derive the vector for this megadocument by applying the standard $tf \cdot idf$ method. Thus, the tf values from the original methods are summed up before the tf weights are computed. In a similar way, the document count for the idf component now refers to the number of megadocuments.

For a given collection, we finally have one megadocument per category, and the total number of megadocuments equals the number of categories. The Yahoo collection comprises 2542 categories, so there are 2542 megadocuments.

[1] `http://www.yahoo.com/`
[2] `http://www.dino-online.de/`
[3] `http://www.research.att.com/~lewis/reuters21578.html`

In Reuters we have 151 megadocuments and in Dino-Online there are 1211 megadocuments.

To classify a new document, first all terms of the document are extracted, and then the n best terms (according to their idf values) are selected and used as query for retrieving the most similar megadocuments. The similarity search is based on the scalar product, which is calculated using the probabilistic deductive database engine HySpirit [Fuhr & Rölleke 98].

2.1 Term weighting

For term weighting of the megadocuments we use the standard tf and idf weights [Salton & Buckley 88]. For every term in a document we calculate the tf with the following formula:

$$tf_{i,d} = 0.5(1 + f_{i,d}/m_d). \tag{1}$$

Here $f_{i,d}$ is the term frequency of the term t_i in the document d and m_d is the maximum term frequency of a term within this document.

The inverse document frequency of an index term is calculated by the formula:

$$idf_i = \log N/n_i, \tag{2}$$

where N is the total number of documents in the collection and n_i is the number of documents which contain the term t_i.

The overall weight calculated by HySpirit is then:

$$w_{i,d} = tf_{i,d} \cdot idf_i, \tag{3}$$

where i denotes the term index and d the document index. It is assumed that terms are mutually independent.

In order to increase efficiency (and possibly effectiveness) of the categorization process, not all terms of a test document are considered for categorization. In order to select the best n terms that are actually used for classification, terms are ranked according to decreasing idf values — assuming that terms with high idf values are good discriminators. Thus, we use only these terms as query terms for retrieving the most similar megadocument.

2.2 Classification

The query vector resulting from a test document is transformed into a probabilistic datalog query ([Fuhr 95]), which is then evaluated with HySpirit. For example, a probabilistic datalog program with two query terms looks like this:

```
category(C) :- term_edb(bedlington,C) & termspace_edb(bedlington).
category(C) :- term_edb(desci,C) & termspace_edb(desci).
?- category(C).
```

The `term_edb` predicate represents the tf weight and retrieves the documents which then are probabilistically multiplied by the idf weight represented by `termspace_edb` predicate.

In order to illustrate the basic idea of our approach, we compare it with the kNN method [Yang 94] (which is used for the same task, as described in

[Gövert et al. 99]). Two different document spaces are shown in figure 2. In the kNN space shown on the left all training documents are represented as vectors in the space. After computing the vector for a test document (shown in the center of the circle), the k-nearest-neighbors (in the circle) are considered. Based on the categories of these documents, the final category assignment is performed.

In the right-hand document-space, megadocuments are visualized as circles (of varying size). Here every circle corresponds to a category and each category occurs exactly once. Given the query vector for the test document, its category is assigned by looking at only one megadocument, namely that with the nearest vector.

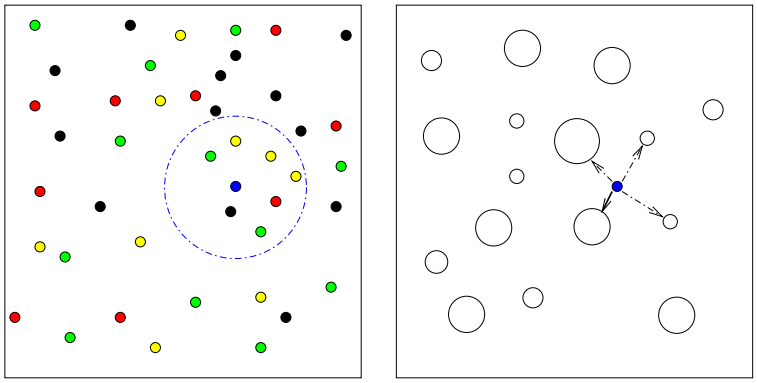

Figure 2: Classification

The classification procedure is fast, because only the number of categories, instead of all documents, have to be compared for retrieving the category to be assigned to a given document.

3 Evaluation settings

For storing the document index of the megadocuments, which can be as large as 9 MB, we use MySQL[4], a relational database.

The construction of the Yahoo and Dino-Online collection was performed by ourselves, with a program written specifically for this purpose. As (test and training) documents, not only the Web pages referenced by the respective catalogs were considered. These Web pages often contain hardly any text — e.g. when they are the root of a frame-set, or consist of a graphics for navigating through the content of a site. Thus, we considered the "radius 1" documents, that is, all documents that have links from the page referenced in the catalog, but only those residing on the same server.

Since both Web catalogs have hierarchic classification trees, we performed two kinds of evaluation: A `top` match between the automatic and the manual

[4] `http://www.mysql.com/`

categorization is assumed if the two assignments agree with respect to the 25 top-level categories in Yahoo (23 top-level categories in Dino-Online). In contrast, an `all` match requires that both documents are assigned to the same node of the classification tree.

For the experiments, the Web collections were considered as a whole, with a 2:1 split between learning and test sample; this split was performed category-wise, thus ensuring that we have training documents for each category.

3.1 Yahoo Collection

The Yahoo collection consists of the complete subtree of "Computers & Internet" as can be seen in figure 3. The tree consists of 7 levels with 2806 categories. The terms per category vary from 1 to 20 904 terms. There are 18 639 documents in the subtree, which are divided into 12 315 learning documents and 6324 test documents. The raw data size is 500 MB, the index within MySQL is 752 MB. The collection consists of textual data only. Images, speech, Java scripts and applets were removed from the Web pages,

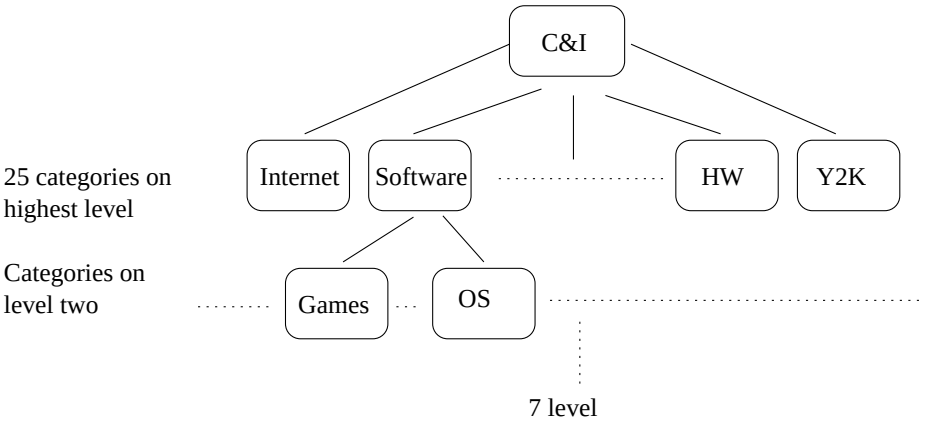

Figure 3: Yahoo tree

In table 1, the results of our experiment and those of two comparable approaches are presented. As effectiveness measure, we consider precision in the first rank. We chose this measure because (with very few exceptions) the manual classification puts each Web page into a single category only. We varied the number of query terms (taken from the document to be classified) from 10 to 50; this restriction of terms showed no negative effect on results. Rather, the number of hits slightly increased.

For comparison, in [Chakrabarti et al. 98] two evaluations are described. Their `baseline`, is a simplistic version of the Rocchio method. The `link based` approach considers only documents which are referenced by other Web pages, which, in turn, have entries in the Yahoo catalog. Based on the knowledge about the categorization of these other Web pages, the document in question can be classified with a relatively high accuracy; however, this procedure is not realistic, since new Web pages should be classified as soon as possible, and not

after several months when there are enough other pages referencing the page in question. In [Gövert et al. 99] the kNN method is used for classification, in combination with a new indexing procedure for Web pages. Overall, our new method surpasses the quality of comparable approaches by about 15 % (absolute difference).

number of terms/approach	match	hits
50	all	12.52 %
50	top	48.30 %
10	all	13.73 %
10	top	51.07 %
[Chakrabarti et al. 98] baseline	13 top categories	32 %
[Chakrabarti et al. 98] link based	13 top categories	75 %
[Gövert et al. 99] kNN	top	36.5 %

Table 1: Evaluation Yahoo Catalog

3.2 Dino Collection

The Dino Collection is a German Web catalog. Here we spidered the whole category tree. From the total of 55 672 documents, we chose 18 577 as test documents. There are 1211 categories distributed over 3 levels. Table 2 shows the evaluation results. The raw data size is 223 MB, the size of the MySQL database is 410 MB.

number of terms	match	hits
10	all	32.09%
10	top	53.47%

Table 2: Evaluation Dino-Online Catalog

The hits on all categories are noticeable better then in the Yahoo evaluation. One possible reason for this improvement could be the simpler structure of the classification tree — three levels in Dino vs. up to seven levels in Yahoo. Another factor may be the fact that we considered the whole catalog of Dino-Online, but only the "Computer & Internet" subtree of Yahoo; thus, the Dino classification scheme is more coarse-grained, and the task may be easier.

Another idea is to generate only the 23 mega-documents for the top-level categories, by folding there sub-levels together. This maybe used to find the right entry into the hierarchy of the collection.

The difference with respect to the `top` result is the change in the termspace. While in the former experiment the *idf* weights have been computed with respect to 1211 mega-documents, this time only 23 mega-documents were used to compute the *idf* weights for the termspace.

The results in table 3 surprisingly show worse result then in the former `top` experiment. With the new termspace we reach a precision in the first rank

number of terms	termspace	hits
10	1211 mega-documents	47.46%
10	23 mega-documents	42.76%

Table 3: Evaluation Dino-Online 23 top-level categories

which is about 5% worse than before.

One explanation for this phenomena could be the granularity of the termspace weighting. Having only 23 different documents makes it probable that many terms share the same *idf* weight.

3.3 Reuters Collection

Since we have developed a new classification method, we also want to get results that are comparable with those of the "classic" approaches. For this purpose, we used the Reuters collection, which has been widely used by other researchers. The collection consists of 12 000 document in SGML format which are already divided into 3 299 test and 9 603 learning sample by Lewis (LEWIS-SPLIT sgml-tag), which he used in his papers (see README.txt of collection or [Lewis & Ringuette 94]). For these experiments, we did not consider any category hierarchy. The raw data size is 27 MB, and the MySQL database is 73 MB.

terms	strategy	hits
10	all	70.22%
[Yang 99]	kNN	85 %
[Yang 99]	Rocchio	65 %

Table 4: Evaluation Reuters

As can be seen in table 4, we reached a precision of 70 %. According to [Yang 99], this figure is higher than that of the simple Rocchio approach. On the other hand, here kNN performs much better than our megadocument method. Comparing these results with those from the Yahoo collection, it turns out that the type of the collection seems to have an important effect on the quality of the classification method. Obviously, the classical categorization methods work well on rather homogeneous collections, but run into problems when faced with heterogeneous documents like e.g. from the Web. Since the results presented in this paper are outcomes of a first study in this area, further work is needed in order to make final statements about the suitability of the different methods for Web categorization.

4 Conclusions and future work

In this paper, we have presented a new approach for classifying Web documents. Our approach reaches relatively good results with Web documents, whereas clas-

sical approaches perform badly for these collections. Furthermore, the classification procedure is relatively simple and can be performed efficiently — due to the fact that the search space for the similarity search contains only one entry per category. With standard collections like Reuters our method achieves a quality that is comparable to the average of other classical methods. In addition, further improvements are possible by fine-tuning our method.

In future work, we will aim at increasing classification accuracy by considering the hierarchic structure of the classification scheme hierarchy. Another area of research is the improvement of the document indexing methods, i.e. by replacing $tf \cdot idf$ weighting by the description-oriented method that has been used in [Gövert et al. 99]; this method is able to consider the structure of documents (e.g. the different tags in HTML documents) for optimizing term weighting.

An interesting outcome of this study is the poor performance of classic categorization methods on Web collections. Further evaluations with other homogeneous and heterogeneous collections have to be performed in order to test the validity of the results achieved so far.

References

Chakrabarti, S.; Dom, B.; Indyk, P. (1998). Enhanced Hypertext Categorization Using Hyperlinks. In: Haas, L.; Tiwary, A. (eds.): *Proceedings of the 1998 ACM SIGMOD. International Conference on Management of Data.* ACM Special Interest Group on Management of Data, ACM, New York. http://www.acm.org/sigmod/sigmod98/eproceedings/.

Fuhr, N.; Rölleke, T. (1998). HySpirit — a Probabilistic Inference Engine for Hypermedia Retrieval in Large Databases. In: Schek, H.-J.; Saltor, F.; Ramos, I.; Alonso, G. (eds.): *Proceedings of the 6th International Conference on Extending Database Technology (EDBT), Valencia, Spain*, Lecture Notes in Computer Science, pages 24–38. Springer, Berlin et al.

Fuhr, N. (1995). Probabilistic Datalog - a Logic for Powerful Retrieval Methods. In: Fox, E.; Ingwersen, P.; Fidel, R. (eds.): *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 282–290. ACM, New York.

Gövert, N.; Lalmas, M.; Fuhr, N. (1999). A probabilistic description-oriented approach for categorising Web documents. In: *Proceedings of the 9th international conference on Information and knowledge management*, pages 475–482. ACM, New York.

Lewis, D.; Ringuette, M. (1994). A comparison of two learning algorthms for text categorization. In: *Symposium on Document Analysis and Information Retrieval.* Las Vegas.

Salton, G.; Buckley, C. (1988). Term Weighting Approaches in Automatic Text Retrieval. *Information Processing and Management 24(5)*, pages 513–523.

Yang, Y. (1994). Expert Network: Effective and Efficent Learning from Human Decisions in Text Categorisation and Retrieval. In: Croft, W. B.; van Rijsbergen, C. J. (eds.): *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 13–22. Springer-Verlag, London, et al.

Yang, Y. (1999). An Evaluation of Statistical Approaches to Text Categorization. *Information Retrieval 1(1)*, pages 69–90.