An unsupervised hierarchical approach to document categorization

Robert Wetzker
Technische Universität Berlin
DAI-Labor
10587 Berlin, Germany
robert.wetzker@dai-labor.de

Tansu Alpcan
Deutsche Telekom Laboratories
10587 Berlin, Germany
tansu.alpcan@telekom.de

Christian Bauckhage
Deutsche Telekom Laboratories
10587 Berlin, Germany
christian.bauckhage@telekom.de

Winfried Umbrath
Technische Universität Berlin, DAI-Labor
10587 Berlin, Germany
winfried.umbrath@dai-labor.de

Sahin Albayrak
Technische Universität Berlin, DAI-Labor
10587 Berlin, Germany
sahin.albayrak@dai-labor.de

***Abstract*— We propose a hierarchical approach to document categorization that requires no pre-configuration and maps the semantic document space to a predefined taxonomy. The utilization of search engines to train a hierarchical classifier makes our approach more flexible than existing solutions which rely on (human) labeled data and are bound to a specific domain. We show that the structural information given by the taxonomy allows for a context aware construction of search queries and leads to higher tagging accuracy. We test our approach on different benchmark datasets and evaluate its performance on the single- and multi-tag assignment tasks. The experimental results show that our solution is as accurate as supervised classifiers for web page classification and still performs well when categorizing domain specific documents.**

I. Introduction

Since the beginning of the Internet there have been efforts to structure its content in order to help people find what they are looking for. One way of structuring content is to categorize documents based on their topics. This is, for instance, done successfully in directory services or tagging systems where users label the content by classifying it into certain categories of a predefined taxonomy or a set of keywords. The success of directory services and tagging systems underlines the importance of web content categorization. However, considering the enormous amount of data on the web a manual categorization of content can only be an initial step and ways of automated document tagging have to be found.

A. Related work

The task of automated text categorization (TC) is a well known information retrieval and machine learning problem concerned with the "labeling of natural language texts with thematic categories" [1]. Hierarchical text classifiers have attracted the attention of the research community (e.g. [2]–[8]) as they have the potential of decomposing the classification task into smaller classification problems, thus being potentially faster and more robust than their non-hierarchical counterparts. In addition, the semantic relations between categories, given by their relative position in the hierarchy, can be used for a more consistent category representation. The authors of [4], for instance, apply a method called *shrinkage* in order to smooth the feature distributions between a category and its ancestors.

Most classifiers described in the TC literature depend on labeled data during the training phase. This not only limits their applicability to fields where labeled data can be provided at cheap costs but also makes the results obtained by these classifiers less universally valid. McCallum *et al.* [5] try to overcome this shortcoming using a bootstrapping approach where each category is initially presented by very few keywords which augment with every newly classified document. This is somewhat similar to our approach in that we use the topic names of the taxonomy as the seed for the query string construction. However, the manual specification of "few keywords" for each topic in [5] seems to make higher demands to the user compared to the assigment of simple topic names as in our approach. Furthermore our approach is somewhat more "exogenous" as the initial population of the taxonomy is based on search engine results and not on domain specific data.

The popularity of directory services such as Yahoo! or the Open Directory Project as well as the upcoming of the Semantic Web concept have increased interest in the TC world for the problem of automatic web page classification. Many classifiers have been proposed and succesfully tested that benefit from web site specific characteristics such as anchor-texts [9] or the link structure [10]. On the other hand the task of web page classification is also often used as an TC benchmark for more general classifiers treating web pages simply as documents as done in this paper. The tests conducted in [4], [6], [7], [11] are very similar to those presented here as they were performed on directory services such as Yahoo! considering the taxonomy and the assigned web pages as ground truth. Results reported by the authors vary from around

0.3 [7] to 0.5 [6] for the microaveraged F1-measure.

The reuters RCV1 corpus [12] of manually categorized newswire stories is a well known TC benchmark dataset. Different classifiers have been proposed and tested on reuters. Some of them were found to be very accurate achieving microaveraged F1-measures above 0.8. SVM and k-NN classifiers have been reported to perform best [12], [13].

B. Our contribution

The fact that we utilize web search for the automated configuration of our classifier makes our approach much more flexible than other solutions which rely on (human) labeled training data. All the supervision needed from the user is the specification of the taxonomy. Furthermore we present the α parameter which makes it easy for the user to control the number of tags that will be assigned to documents. As our solution accesses search engines to determine the relevant features for each category it is not bound to any specific domain but generally applicable.

Moreover, we show that the structural information contained within the taxonomy does not only allow for the construction of context aware search queries during the feature generation phase but can also increase accuracy when incorporated in hierarchical classifiers. We demonstrate that feature propagation and levelwise probabilistic matching can help in constructing hierarchical classifiers that outperform their flat counterparts.

II. Populating the taxonomy

In the area of Knowledge Management taxonomies generally refer to trees of topics where the child topic is a specialization of the parent topic. The only human supervision needed for our algorithm is the specification of the taxonomy. This can simply be done by specifying a set of relevant topics, given by a representative topic string e.g. "Parallel Computing", and the context given by the position in the taxonomy e.g. as child of the "Computers" node.

Once the taxonomy is defined we query the Yahoo! Search Webservice[1] in order to create a representative *bag-of-words* for each category. The query is constructed using the label assigned to each category and the label of the parent category as query keywords, e.g. for the node "Artificial Intelligence" in the "Computers" branch the query string would be "Computers Artificial Intelligence". By including the parent category we maintain the structural information given by the position of a category in the taxonomy thus allowing for a context aware feature generation. We limit the search results to English web pages only by applying the "format" and "language" filters provided by the API. The web service returns a ranked list of the most related web pages and their URLs. The first N websites of the result set are downloaded and processed, removing all HTML tags, in order to extract all contained terms. Stopwords are removed and all terms are stemmed using the Porter stemming algorithm [14]. Following [12] we then weight each feature by tfidf:

$$w_d(t) = (1 + \log_e n(t,d)) * log_e(|D|/n(t)) \qquad (1)$$

where n(t) is the number of overall documents that contain term *t*, n(t,d) is the number of occurences of term *t* in the document *d* and $|D|$ being the total number of documents. Afterwards we normalize each document's feature vector to the Euclidian norm of 1.0. The feature vector for each category is created by aggregating the feature vectors of its documents and normalizing the resulting vector to unit length.

For the hierarchical case we additionaly propagate the feature distributions of all categories from the leafs towards the root node. This way we ensure that the structural information given by the taxonomy is represented in the feature vector space. The feature propagation is realized by recursively aggregating all children feature weights to the parent as shown in (2):

$$w'_c(t) = w_c(t) + \sum_{i \in children(c)} w_i(t) \qquad (2)$$

After each propagation step we once again normalize the resulting parent feature vector to unit length as otherwise branches containing more categories would be overestimated.

III. The hierarchical classifier

The tagging algorithm decides what categories to assign to a given document. This decision problem can be equally interpreted as choosing the subgraph of the taxonomy that best represents the given document.

Before the matching itself starts, we preprocess the given document the same way we did for the training documents by extracting all terms, removing stopwords, stemming and applying *tfidf*.

The levelwise classification follows the method presented in [15] and is described in Figure 1. The algorithm maps a given document to a subtree of the taxonomy. All categories of the subtree are then considered valid tags for this document. Similarities are calculated using the cosine distance between the feature vectors of the given document and a category. The α parameter directly influences the number of tags assigned to a given document. Setting α to a very high value makes the algorithm choose only a single path. Thus, according to his taste, the user can easily influence whether few or many tags should be assigned to documents without fixing the number of tags. Step (6) of our tagging algorithm allows that documents may also be classified into internal nodes which is an advantage above other hierarchical classifiers such as the one presented in [4].

The non-hierarchical classifier is trained exactly the same way as the hierarchical except that no features are propagated. In order to classify a given document it calculates the similarity to all categories and assigns the most similar N categories and all their ancestors as labels.

[1] http://developer.yahoo.com/search/

Input: document $\mathcal{D}$ to categorize, taxonomy $\mathcal{T}$, similarity measure sim
Output: corresponding subtree categorizing document $\mathcal{D}$

/ Find the subtree of the taxonomy $\mathcal{T}$ that best categorizes a given document $\mathcal{D}$ iteratively from top to bottom */*
1. set the root node as *current node*
2. compute the similarity of the given document $\mathcal{D}$ to all child nodes of the *current node*
3. compute the mean μ and the standard deviation σ of the resulting similarities
4. consider all nodes $\{n_k\}$ in the current set of children for which $sim(\mathcal{D}, n_k) > \mu + \alpha\sigma$, where α is a fixed parameter
5. if $\{n_k\} = \emptyset$, choose the most similar child as $\{n_k\}$
6. if $max(sim(\mathcal{D}, n_k)) < sim(\mathcal{D}$, *current node*), add the *current node* and its ancestors as valid tags and stop
7. for each node n_k in the set $\{n_k\}$: if n_k is a leaf node, add n_k and all ancestors to the list of valid tags, else set n_k as *current node* and continue with step 2

Fig. 1. Algorithm to map a document characterized by its feature vector to a subtree of a category taxonomy. All nodes of the resulting subtree are considered valid document tags.

IV. Experiments

For the evaluation of our approach we set up two different test scenarios. In the first scenario we determine the performance of our classifier in web page classification where each web page appears in one taxonomy branch only. In the second scenario we use the Reuters benchmark corpus [12] to show that our method is not restricted to web pages but also performs well when dealing with other types of text.

A. Web page classification

For the experiments reported here we choose a taxonomy based on the category tree of the Open Directory Project (ODP)[2].

The taxonomy extracted from the ODP contains all first and second level topics resulting in 14 first-level and a total number of 411 nodes. We only excluded the first level "World" branch as it consists of mostly non english content. In order to represent each topic node by a feature vector we follow the steps described in section II.

We treat all web pages assigned by web users to the different ODP topics as ground truth and consider all categories on the path to each web page as tags. From the 4283 documents assigned to the 411 topics of the taxonomy we remove the documents that were also part of the crawled training data and use the remaining documents for testing. The test data is not uniformly distributed over the topics which we consider a realistic scenario.

As all the web pages of the test set appear in one taxonomy branch only, we set the α parameter of the hierarchical classifier to ∞. Thus the classifier just chooses the most promising branch.

B. Reuters corpus

The Reuters corpus version RCV1 [12] consists of over 800.000 newswire stories manually categorized by experts. We choose Reuters as it is a well known text categorization benchmark collection and provides a corpus of multi-tagged documents which lets us investigate the performance of our method when assigning multiple tags. Furthermore, using reuters we can test whether our method is effective when confronted with non web site texts.

Reuters uses three different tag sets for the categorization of news entries (*topics*, *industries*, *regions*). As the tags from the *regions* and the *industries* sets are not (or badly) structured, not providing any useful hierarchy [12], we only consider the 103 *topics* tags in our evaluations. These are complete in the sense that all ancestors of a valid category are valid, too.

We polpulated the given *topics* taxonomy the same way as in our web page classification scenario using always the highest ranked 20 web sites for the vocabulary creation. As the test set we choose the 23418 newswire entries that were used for training the classifiers in [12].

V. Results

Table I compares the micro- and macroaveraged precision and recall results obtained by our hierarchical classifier on the web site categorization task to the performance of the non-hierarchical version. The results are given for different numbers of web sites used during the initial vocabulary generation. As can be seen from both tables using more web pages does not necessarily increase the result quality. Instead, taking only the best ranked 20 web pages already proves sufficient. This observation holds for the hierarchical as well as for the non-hierarchical classifier and is valid for the macroaverage and the microaveraged case. The non-hierarchical classifier is more sensitive to the number of web pages, especially in the macroaveraged case, whereas feature propagation has made the hierarchical classifier more robust. The fact that few pages achieve better results is important, since, without any feature reduction, the number of web pages used for training directly correlates to the number of features representing each topic and thus influences classification speed.

The non-hierarchical classifier receives a significantly higher accuracy on the web site classification challenge outperforming the tree classifier by up to 9%. This indicates that our hierarchical method is less accurate on the first level. The information that a leaf category has the highest similarity to

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

TABLE I

OPEN DIRECTORY (411 CATEGORIES)- HIERARCHICAL VS. NON-HIERARCHICAL PRECISION AND RECALL RESULTS

(a) microaveraged

# training websites	Hierarchical Classifier			Non-hier. Classifier		
	Prec.	Rec.	F1	Prec.	Rec.	F1
10	0.562	0.454	0.502	0.582	0.575	0.578
20	0.551	0.475	0.510	0.600	0.595	0.597
50	0.535	0.489	**0.511**	0.606	0.599	**0.603**
100	0.516	0.486	0.501	0.593	0.587	0.590

(b) macroaveraged

# training websites	Hierarchical Classifier			Non-hier. Classifier		
	Prec.	Rec.	F1	Prec.	Rec.	F1
10	0.383	0.239	0.294	0.357	0.336	0.346
20	0.374	0.241	0.293	0.359	0.395	**0.376**
50	0.369	0.250	0.298	0.360	0.341	0.350
100	0.362	0.256	**0.300**	0.360	0.343	0.351

a given document seems not to be sufficiently represented by the feature propagation algorithm. This still has to be further investigated. Especially the single-tagging case with $\alpha = \infty$ seems to be affected.

It is difficult to compare the effectiveness of our classifier to other reported results as test setups vary throughout the literature. To our knowledge, the test setups most similar to ours can be found in [6] and [7]. The authors evaluate the effectiveness of their models based on subsets of the Yahoo! directory service which is akin to our evaluation on the ODP taxonomy. [7] report maximal (microaveraged) F1 measures ranging from 0.47 to 0.53 for different subbranches of the Yahoo! hierarchy whereas the maximal F1 measure reported by [6] is 0.497. Both our hierarchical and non-hierarchical classifier seem to classify at least as good as these other methods with the non-hierarchical classifier beating the best reported value by 7%. This is especially interesting as the other classifiers where trained on labeled domain data which should increase their effectiveness.

Figure 2 visualizes the effectiveness of the hierarchical and the non-hierarchical classifier on the reuters *topics* data. For the hierarchical case the precision and recall values are given for various α parameters. For the non-hierarchical case the values are given for different fixed numbers of tags assigned to each document. Figure 2(a) shows that the α parameter gives easy control over the amount of tags enabling the user to trade precision for recall. Still the hierarchical classifier does not restrict the user to fix the number of assigned tags as is the case for the non-hierarchical classifier.

Table II presents the parameters that optimize the F1 values for reuters. For the microaveraged case F1 is maximized choosing an α value of 0.2 whereas an α of 0.0 optimizes the macroaveraged F1. On average the hierarchical classifier identified 1.56 (1.66) of the tags manually assigned to a document. For the non-hierarchical classifier it is always best to label with two tags and their ancestor categories.

In contrast to the results obtained on the ODP data, the hierarchical classifier proves superior on the task of multi-tagging and achieves better microaveraged results than the non-hierarchical model. This probably results from the fact that the hierarchical classifier uses a stochastical model in order to decide how many tags to assign to a given document. This makes it more flexible compared to the non-hierarchical classifier that fixes the number of tags. However, the non-hierarchical classifier still performs better when averaged over all categories.

Existing studies in the literature report microaveraged F1 values above 0.8 on the reuters *topics* data with SVMs and k-NN classifiers performing best [12], [13]. As expected, our approach does not reach the same accuracy as these specialists. However, given the generality of our approach, we consider a F1 value of 0.52 a good result. The fact that we do not train our classifier on domain specific data leads to more noisy training data and makes it impossible to catch any bias possibly contained in the test data. Also, the results reported by other authors are only meaningful for situations where a huge amount of labeled data can be provided at cheap costs.

VI. CONCLUSIONS AND FUTURE WORK

We have proposed a novel method for hierarchical document categorization that needs no supervision during the training phase. Our approach is more flexible than many domain specific solutions and can be configured and calibrated by simply specifying a taxonomy and setting a single parameter. This simplicity is certainly an advantage over other methods such as SVMs which demand much effort during the configuration phase.

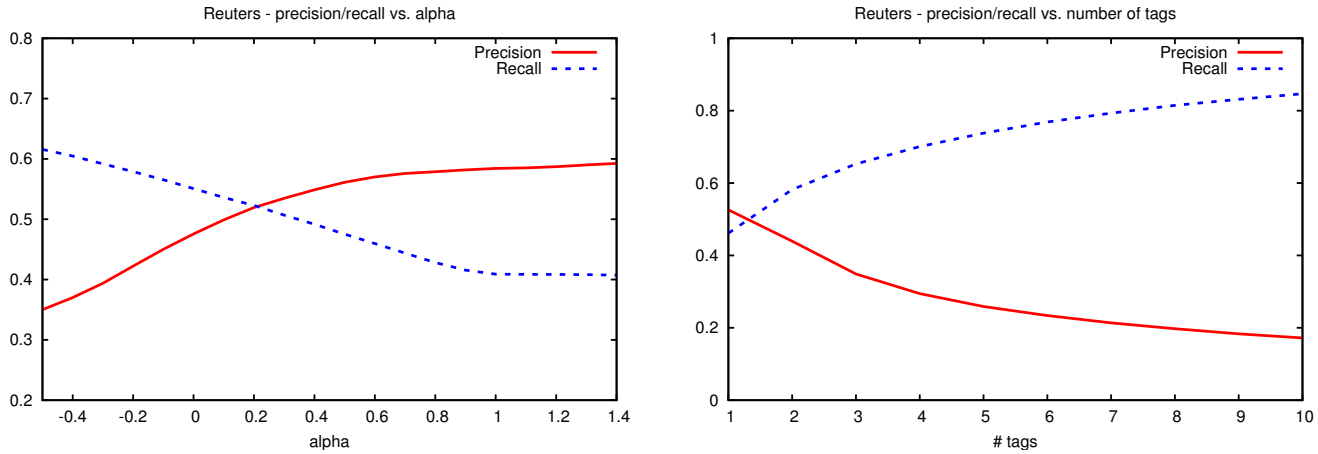
We have shown that the semantical information contained in the taxonomy structure allows for a context aware construction of search queries. Moreover, we have provided empirical evidence that feature propagation helps in constructing hierarchical classifiers that are more accurate than non-hierarchical versions on the task of multi-tagging.

The *bag-of-words* approach is limited to the information contained in the documents themselfs. To overcome this limitation we plan to investigate the effects of feature enrichment using publicly available background knowledge. Sources could once again include search engines, but also the results reported for WordNet [16], Directory Services [17] and Wikipedia [18] seem very encouraging.

The results reported by other authors indicate that many features are not needed for a successful categorization. Instead, removing more than 90% of all features showed no or only very small negative effect on the accuracy [3], [13]. Therefore, in addition to feature enrichment, we are going to examine, how feature reduction can be added to our approach in order to increase its effectiveness and speed. Preliminary results in this direction are very promising.

REFERENCES

[1] F. Sebastiani, "Machine learning in automated text categorization," *ACM Comput. Surv.*, vol. 34, no. 1, pp. 1–47, 2002.



(a) hierarchical (b) non-hierarchical

Fig. 2. Recall and precision results on the reuters *topics* test set for various α parameters in the hierarchical (a) and various numbers of selected tags in the non-hierarchical (b) case. For the non-hierarchical results ancestors of a selected tag are automatically considered valid tags in order to make the results comparable.

TABLE II

REUTERS - HIERARCHICAL VS. NON-HIERARCHICAL PRECISION AND RECALL RESULTS FOR BEST PARAMETER SETTINGS

averaged by	**Hierarchical Classifier**					**Non-hier. Classifier**			
	α*	Tags	Prec.	Rec.	F1	Tags*	Prec.	Rec.	F1
micro	0.2	1.56	0.519	0.523	**0.521**	2	0.439	0.582	**0.501**
macro	0.0	1.66	0.248	0.286	**0.266**	2	0.227	0.483	**0.323**

[2] S. Chakrabarti, B. Dom, R. Agrawal, and P. Raghavan, "Scalable feature selection, classification and signature generation for organizing large text databases into hierarchical topic taxonomies," *The VLDB Journal*, vol. 7, no. 3, pp. 163–178, 1998.

[3] D. Koller and M. Sahami, "Hierarchically classifying documents using very few words," in *ICML '97: Proc. of the 14th Int. Conf. on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997, pp. 170–178.

[4] A. McCallum, R. Rosenfeld, T. M. Mitchell, and A. Y. Ng, "Improving text classification by shrinkage in a hierarchy of classes," in *ICML '98: Proc. of the 15th Int. Conf. on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, pp. 359–367.

[5] A. McCallum, K. Nigam, J. Rennie, and K. Seymore, "A machine learning approach to building domain-specific search engines," in *IJCAI '99: Proc. of the 16th Int. Joint Conf. on Artificial Intelligence*, 1999, pp. 662–667.

[6] S. Dumais and H. Chen, "Hierarchical classification of web content," in *SIGIR '00: Proc. of the 23rd annual int. ACM SIGIR conf. on Research and development in information retrieval*. New York, NY, USA: ACM Press, 2000, pp. 256–263.

[7] D. Mladenic and M. Grobelnik, "Mapping documents onto web page ontology." in *EWMF*, ser. Lecture Notes in Computer Science, vol. 3209. Springer, 2003, pp. 77–96.

[8] X. PENG and B. CHOI, "Automatic web page classification in a dynamic and hierarchical way," in *ICDM '02: Proc. of the 2002 IEEE Int. Conf. on Data Mining (ICDM'02)*. Washington, DC, USA: IEEE Computer Society, 2002, p. 386.

[9] E. J. Glover, K. Tsioutsiouliklis, S. Lawrence, D. M. Pennock, and G. W. Flake, "Using web structure for classifying and describing web pages," in *WWW '02: Proc. of the 11th int. conf. on World Wide Web*. New York, NY, USA: ACM Press, 2002, pp. 562–569.

[10] G. Attardi, A. Gullí, and F. Sebastiani, "Automatic Web page categorization by link and context analysis," in *Proc. of THAI-99, European Symposium on Telematics, Hypermedia and Artificial Intelligence*, Varese, IT, 1999, pp. 105–119.

[11] C.-C. Huang, S.-L. Chuang, and L.-F. Chien, "Liveclassifier: creating hierarchical text classifiers through web corpora," in *WWW '04: Proc. of the 13th int. conf. on World Wide Web*. New York, NY, USA: ACM Press, 2004, pp. 184–192.

[12] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li, "Rcv1: A new benchmark collection for text categorization research," *J. Mach. Learn. Res.*, vol. 5, pp. 361–397, 2004.

[13] M. Rogati and Y. Yang, "High-performing feature selection for text classification," 2002. [Online]. Available: citeseer.ist.psu.edu/rogati02highperforming.html

[14] M. F. Porter, "An algorithm for suffix stripping," *Program*, vol. 14, no. 3, pp. 130–137, July 1980.

[15] T. Alpcan, C. Bauckhage, and S. Agarwal, "An efficient ontology-based expert peering system," in *Proc. of the 6th IAPR Workshop on Graph-based Representations in Pattern Recognition*, June 2007.

[16] A. Hotho, S. Staab, and G. Stumme, "Wordnet improves text document clustering," in *Proc. of the SIGIR 2003 Semantic Web Workshop*, Toronto, Canada, 2003.

[17] E. Gabrilovich and S. Markovitch, "Feature generation for text categorization using world knowledge," in *Proc. of The 19th Int. Joint Conf. for Artificial Intelligence*, Edinburgh, Scotland, 2005, pp. 1048–1053.

[18] E.Gabrilovich and S.Markovitch, "Overcoming the brittleness bottleneck using wikipedia: Enhancing text categorization with encyclopedic knowledge," in *Proc. of the 21st National Conf. on Artificial Intelligence*, 2006.