Supporting Navigation in Large Document Corpora of Online Communities: a Case-Study on the Categorization of the English Wikipedia

Viktor Gál, Domonkos Tikk

Dept. of Telecommunications and Telematics
Budapest University of Technology and Economcs
Magyar tudósok krt. 2, H-1117 Budapest, Hungary
wiking@maeth.com, tikk@tmit.bme.hu

György Biró

TextMiner Co.
Gyulai P. u. 37, H-1029 Budapest, Hungary
george.biro@gmail.com

Abstract: In this paper we present a categorization-based method for supporting navigation in large document corpora that are created, collected, revised, tagged etc. by independent individuals forming an online community. Users often face the problem of redundancy when intend to contribute to the content created by a community. Since different people have different skills and background use of terminology, one may easily 'reinvent the wheel' when adding seemingly new, but semantically duplicated content to an opened document corpus. This reduces the usability and coherence, hence decreases the quality of the corpus. One way to minimize the risk of such problem is to support the users with navigational and searching facilities in the corpus. Our paper proposes to exploit any available topical structure to build a category-based model of the corpus, and to apply categorization for new contents. We performed a case-study on the English Wikipedia, one of the largest online document corpora, which shows that our methodology can be useful for regular users, as well as for administrators. The category-based model was build by the HITEC document processing and categorization framework.

Keywords: Categorization, Wikipedia, HITEC

1 Introduction

1.1 Motivation and Related Works

People always fear of the authenticity of the content created by online communities. Most people do not like to rely on information created by unknown individuals, as they are afraid that it could be created by people who might have not the adequate knowledge in the topic or they intentionally try to mislead the public. This situation holds even if the size of the community has reached a critical level, where self-censorship has been developed (as in case of Wikipedia), because of the various people that edit and create content.

On the other hand, the number of online communities is growing rapidly (e.g. Last.fm the community driven music categorization database or del.icio.us, the social bookmarking site), since in most of the cases Internet users have found out that on such sites they can get relevant information much faster, and more conveniently. At last but not least to obtain the knowledge that we are looking for community created categorization is much more detailed and closer to the way of human thinking than e.g. the keyword-based querying and information retrieval using a search engine. Especially in the case when we do not know where to start the search within a special subject domain.

Some of the contents that created by online communities have grown so big that not only adding new content is difficult without redundancy, but navigating within these worlds became very time-consuming. Since all the content and relations between them is mostly done by the community human error comes into the picture. For instance, some of the important connections among the contents are not being resolved or just simply got forgotten. Therefore, there is a need for decision support systems that could help content creators to have a more precise description for their contributions, and users to navigate within this search space.

Lately, there has been a growing amount of studies that try to use the corpora created by online communities as a training dataset for categorization and identification of objects in a specific domain. For example, Meyer et al. [1] reported on applying Wikipedia's corpus as a training dataset for categorizing Learning Objects into subject categories.

1.2 Hierarchical Text Categorization

In the age of Internet everyone can experience the truth of the famous saying: ‘we are drowning in data but starving for information’. The amount of available data can also degrade the perspicuity of the content brought together by independent individuals of global communities. A common way to manage complexity is using

a hierarchy[1], and text is no exception [2]. Therefore a possible way to cope with the mentioned problem is introducing a structure on the content, and organizing the documents into it. While this task is hardly feasible for the entire content of the internet – though there are some initiatives to categorize the internet, see the Open Directory Project of DMOZ[2] – it is amenable on smaller document collections, such us a document corpus created for a given task by the global community.

Sorting documents into a hierarchical category system (termed taxonomy) can be performed by means of automatic hierarchical text categorizers. In this regard to first papers were published in the late 1990s [2, 3, 4, 5]. However, most of the early methods were not able to cope with extremely large taxonomies and document corpora, such us e.g. the International Patent Classification (IPC) taxonomy consisting of about 5000 categories at the top four levels, since they did not incorporate the hierarchy in the categorization algorithm, but applied classical 'flat' classifier algorithm on the flattened category system. In previous works [6, 7, 8], we have reported on a 'real' hierarchical text categorizer, called HITEC, that adapts its classification algorithm to the given hierarchical category system. The categorization engine HITEC serves as the core of the HITEC document processing and classification framework, which is applied in our experiments. The framework supports various search modes in addition to the classification.

This paper is organized as follows. In Section 2 we introduce the architecture of HITEC framework. Section 3 describes the customization process of HITEC for categorizing Wikipedia articles. Section 4 reports on the results of the categorization of the English Wikipedia corpus, and finally Section 5 concludes the paper and sets out the direction of future works.

2 The HITEC Document Processing and Classification Framework

2.1 System Architecture

HITEC framework[3] is a multi-lingual modularized text management toolset that is able to process, index and categorize text documents. The system consists of

1 Hierarchy is considered to be a directed acyclic graph (DAG).

2 www.dmoz.org

3 The current version of HITEC, HITEC3 is freely available online under GPL license at: http://categorizer.tmit.bme.hu/trac. All specifies files and data formats referred in this paper can be found on above website using the *Browse Source* menu.

numerous independent modules that are connected via the communication interfaces. The typical document processing workflow is depicted on Figure 1.

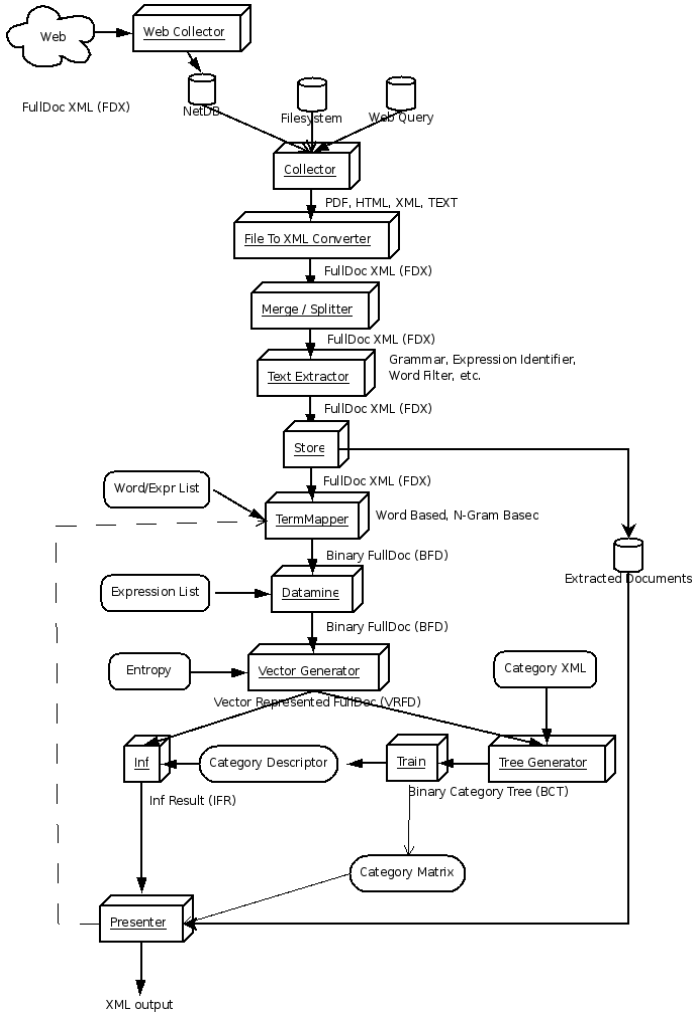

Figure 1
Document processing workflow of the system

From classification point of view we differentiate two document types: training and test. Training documents are endowed with category labels that are used for generating the model, while test documents are not excluded from the building of the model. Test documents also may have label, in which case they can be used to evaluate the performance of the model.

2.2 Data Formats

The system uses its own XML based representation for documents, which is defined in **fulldoc.dtd**. Converters are implemented for the frequent document formats, hence the system can process HTML, PDF, DOC, RTF and plain text files via the converters. This internal format is suitable to store documents in different processing phases, and it is particularly optimized for text mining tasks. This format can be easily converted to any standard XML document schemas (such as, e.g., DocBook, NewsML, TEI[4]).

The **fulldoc** XML is used as input and output format for the text processing modules. The XML format has several dialects related to the different processing phases:

- *raw text*: only structural information is given.
- *token based*: the text is stripped for tokens, sentences are identified if not defined a priori, but no grammatical information is given.
- *grammar extended*: tokens are labeled with grammatical information.

For more information about the DTD see e.g. [9].

Textual representation of documents is less efficient than numerical one for various statistical and data mining algorithm, because string operations are much slower than integer operations. The HITEC framework therefore uses two numerical representations. The BFD format is a numerical id-based sequential image representation of the documents. This is particularly suitable to perform frequent word sequence mining algorithms (cf. [10]) and statistical filters. The VRFD format is a special double bag-of-words representation of documents, since it stores a tf-idf type vector for indexing and full-text search and an entropy-based vector for categorization (see e.g. [11, 12]), and also administrates the category information of the document. The two vectors differ not only in the weight calculation scheme but also in size, since for full-text search all terms of the documents are present in the vector, while for categorization a good portion of irrelevant terms are filtered from the vector space model.

2.3 Modules

After XML converters the **Merger** module merges XML documents into a large common XML file and unifies the character encodings of merged documents. The **Splitter** module splits the merged document collection to document chunks of given size. Both modules use raw text fulldoc dialect as input and output as well.

[4] http://www.docbook.org/schemas/ http://www.newsml.org/pages/spec_main.php, http://www.tei-c.org/P4X/ST.html

The **Text Extractor** component comprises of several submodules that perform various natural language processing tasks:

- **Word Extractor:** This module requires raw text fulldoc dialect as input and outputs token-based fulldoc dialect. It tokenizes the raw text input, i.e. identifies the units of term based processing. If the raw text format does not contain sentence boundaries, it also segments the text into sentences.
- **Stemmer**: The system contains a general stemmer implementation that can be instantiated for the language of the source documents. Currently the module directly supports 15 languages: 15 languages based on the Snowball package [13] (including English, German, French, etc.) and the full-fledged Hungarian stemmer HunStem of the HunMorph package [14]. The module requires token-based fulldoc dialects and generates grammar-extended fulldoc dialect XML as output.
- **Word filter**: This component marks various classes of tokens that are neglected for categorization. It is able to mark closed and open token classes. It works in the former case based on given token lists, and in the latter case based on patterns encoded in recognition rules. For instance it marks general stop-words based on a given language-dependent word list, and numeric tokens, non-alphabetic character sequences using patterns.

The **Store** module assigns the available meta information to each document, such as the category code (primary and secondary codes can be specified) for training document, the URL of different version of the document (original format, raw text and grammar-extended fulldoc format). and other information to document, if available. It works for all fulldoc dialects.

After this stage documents are converted into sequential numerical representation (BFD format) by the **Tokenizer** component. It requires token-based or grammar-extended fulldoc dialect as input and outputs the BFD format of documents. This module operates in daemon mode and is also responsible to manage the dictionary. It determines the identifier of a token if present in the dictionary and assigns a new token identifier to unseen tokens, and inserts them into the dictionary. It is also able to handle multi-word tokens.

The **Vector Generator** module generates VRFD from BFD format. As mentioned in Section 2.2 VRFD format comprises of two vectors: one for indexing (this includes every tokens) using tf-idf weighing, and one for categorization (where irrelevant tokens are excluded) using entropy-weighting [11, 12]. In case of the latter vector the module also applies statistical term filters, such as document frequency based and information-theoretic based term selecting methods [15].

The **Tree Generator** module creates the taxonomy from the category definition file that is given in the form specified in **category.dtd**, and assigns the training documents to the appropriate node of the taxonomy. For this it uses the

categorization vector of the VRFD format to produce a binary category tree (BCT) format as output.

The **Train** module is the core of the entire HITEC framework that generates the classification model. It uses the output Tree Generator module (BCT file) and creates category descriptors. For more information on the internal operation of the Train module see e.g. [6, 7, 8].

The **Inf** module predicts the categories of test documents in form of a weighted category list. The weights can be considered as confidence degree of the prediction. It requires VRFD format input and uses the same as output. It inserts the result of its prediction into the output vector.

The **Indexer** module is applied for full-text search and it determines the documents that match a given query. The queries are processed as (short) test document in the document flow. It requires VRFD as input format and generates the binary Indexer Result (IR) as output.

Finally, the **Presenter** module, which is also used for full-text querying application, displays the category and keyword information of matching documents of the query in **result2.dtd** format, which can be processed by the user interface of the search application. Here the binary input from the Indexer is converted back to text format.

3 Case Study: Supporting Document Processing in Wikipedia

3.1 On Wikipedia

Wikipedia is perhaps the biggest online encyclopedia that is created by individuals around the world. Currently it has almost about 9 million different articles in 253 different languages created by about 9.5 million registered users. Wikipedia's knowledge base is growing rapidly. Since it's start in 2001, the number of articles has been increased by 487 percent on average in a year.

Wikipedia tries to aggregate all the knowledge of mankind in a way that it trusts people with it's content. Anybody can add and edit articles. The administrators (at this moment there are 4166 registered administrators) and other users of Wikipedia are responsible that the articles reflects the truth about the topic.

Recently, Wikipedia with its vast amount of knowledge-base had become more and more important in acquiring new, detailed knowledge in a specific topic. If users try to find facts and knowledge about a certain subject or even if they do not know exactly what are they looking for just have an idea in which area it could

reside they probably ends up browsing between Wikipedia articles. As a matter of fact, even the search engines like Google, MSN and Yahoo direct the users to Wikipedia articles, when they are trying to find exact definitions and facts in a certain theme.

We have chosen to work with the English version of Wikipeida, because this is the biggest monolingual corpus of the Wikipedia currently comprising of about 2 million articles. In addition to that it not only assures a sufficiently lathe amount of data for training and testing, but due to its popularity as a benchmark corpus, it offers an opportunity to compare our works with HITEC and the results with the ones of other authors (see e.g. [1] and [16]), who also studied the problem of extracting different categorization-based information from Wikipedia.

3.2 Category System of Wikipedia Articles

As Wikipedia's database[5] had grown rapidly the maintenance of cross-links (‘See also’ section of the articles) between the articles – that helps users to find related articles in their topic of interest – became very difficult. The problem was not only the maintenance of this list. The navigation within a theme had become more complicated and less efficient for the users in finding their real interest that might have even slightly altered during reading and article. For example if one is interested in Vincent van Gogh's life and work just simply opens the article of van Gogh in Wikipedia. While reading his biography one might be interested about ‘other artists that committed suicide’, ‘other post-impressionist painters’ etc.

To support this kind of a navigation among Wikipedia's articles the users of Wikipedia have introduced categories (e.g. Mathematics, Arts, Flower artists etc.), that is – like everything – maintained by the users themselves. The categories are organized in a hierarchical structure, so one can walk along these categories and find the topic that is the most related to their interest.

Since the categories are maintained by the users there is no guarantee that the category graph is a tree or even that it has no directed circles in it, even though the administrators of Wikipedia, and guidelines of categorization of an article are emphasising that users shall avoid creating circles among categories. In order to guarantee a certain quality of the categorization of the documents, we first had to extract a taxonomy graph from the initial graph. We first built up the taxonomy graph given the links among the categories, and removed those very few edges in the graph that induced circles in it. After this we have chosen the 8 top categories as the roots of the taxonomy graph and applied a breath-first search with a limit of 5000 nodes, hoping that this will cover the most interesting and rich categories of Wikipedia. This initial taxonomy covered about 7% of all the categories in the graph.

[5] http://meta.wikimedia.org/wiki/List_of_Wikipedias

3.3 Creating the Training and Testing Dataset

The publicly available Wikipedia database dump is just one big XML file that contains all the articles, categories, discussions etc. in WikiMedia's template format. In order to extract the category hierarchy and each article's category tag we have first applied WikiPrep [17], a preprocessor for WikiMedia's database dump. It creates a better parseable XML of the database dump, so it eases the translation of the articles from WikiMedia's template system to the fulldoc format.

After transforming all the articles of Wikipedia to fulldoc format, we have decided to have a uniformly distributed knowledge of Wikipedia's knowledge-base, and we have randomly selected 10% of the articles as the training dataset for HITEC. As for the testing dataset we have again randomly selected ten-thousand articles of the remaining 90%. So this way the results of the tests would show us the overall performance of HITEC for categorizing any arbitrary article of Wikipedia. This way we could even see, if HITEC could predict a new category for a given article that might be missing due to the mistakes made by the users.

4 Results

For measuring the quality of the categorization prediction of HITEC we are using the Top, Top-3 and Any measurement quantities [7, 18]. Since in the very corpus there only main categories are available, and no secondary categories, Top and Any measures the same.

We tested the model built by HITEC with 10000 random test documents. Due to parsing errors HITEC was able to predict categories for 8177 of them. 1391 documents were correctly classified by HITEC (17%) in terms of Top (Top 3) measures, i.e. the best prediction (top 3 predictions) of HITEC hit a main category of the article.

At first glance one might think that these results are not impressive at all. One shall not forget the fact that at this point HITEC models only 7% of the entire Wikipedia-taxonomy. If we only consider those 60 test documents that have at least one category in our limited taxonomy from the remaining 6787 articles, we obtained that the Top (Top3) measure of HITEC's prediction is 95.86% (1391/1451). This performance is even more remarkable, if we take into account that there are quite limited number of training documents/categories available (in average 13.72 docs/category).

5 Summary

In this paper we have illustrated how a categorization engine can support the navigation, search and maintenance of large online document corpora created by diverse group of individuals. For this purpose we have applied the HITEC document processing and categorization framework in a case study on the English article corpus of Wikipedia. We have demonstrated that such a supporting tool can improve the quality of the content by e.g. filtering redundant articles.

As for the future we are planning to extend our limited taxonomy tree of Wikipedia to cover all its categories to enable HITEC to be able to predict within the whole subject domain of Wikipedia.

Acknowledgement

Domonkos Tikk was supported by the János Bolyai Research Scholarship of the Hungarian Academy of Science.

References

[1] M. Meyer, C. Rensing, R. Steinmetz: Categorizing Learning Objects Based on Wikipedia as Substitute Corpus. *Proc. of LODE-07, the 1st Int. Workshop on Learning Object Discovery & Exchange*, September 2007

[2] S. Chakrabarti, B. Dom, R. Agrawal, P. Raghavan. Scalable Feature Selection, Classification and Signature Generation for Organizing Large Text Databases into Hierarchical Topic Taxonomies. *The VLDB Journal*, **7**(3):163-178, 1998

[3] D. Koller, M. Sahami. Hierarchically Classifying Documents Using a Very Few Words. *Proc. of ICML-98, the 14th International Conference on Machine Learning*, Vol. 14. San Mateo, CA: Morgan-Kaufmann, 1997

[4] S. D'Alessio, K. Murray, R. Schiaffino, A. Kershenbaum. The Effect of Using Hierarchical Classifiers in Text Categorization. *Proc. of RIAO-00, the 6th International Conference Recherche d'Information Assistee par Ordinateur*, Paris, France, 2000, pp. 302-313

[5] W. Wibovo, H. E. Williams. Simple and Accurate Feature Selection for Hierarchical Categorisation. *Proc. of DE-02, ACM Symposium on Document Engineering*, McLean, Virginia, USA, 2002, pp. 111-118

[6] D. Tikk, Gy. Biró. Experiments with Multi-Label Text Classifier on the Reuters Collection. *Proc. of ICCC 2003, the IEEE Int. Conf. on Computational Cybernetics*, pp. 33-38, Siófok, Hungary, 2003

[7] D. Tikk, Gy. Biró, J. D. Yang. Experiments with a Hierarchical Text Categorization Method on WIPO Patent Collections. In N. O. Attok-Okine and B. M. Ayyub, editors, *Applied Research in Uncertainty Modelling and Analysis*, number 20 in Int. Series in Intelligent Technologies, pp. 283-302, Springer, 2005

[8] D. Tikk, Gy. Biró, A. Törcsvári. A Hierarchical Online Classifier for Patent Categorization. In H. A. do Prado and E. Ferneda, editors, *Emerging Technologies of Text Mining: Techniques and Applications*. Idea Group Inc., 2007

[9] D. Tikk, Gy. Biró, F. P. Szidarovszky, Zs. T. Kardkovács, M. Héder, G. Lemák: Categorization-based Topic-oriented Internet Search Engine. *Proc. of the 7th Int. Symp. of Hungarian Researchers on Computational Intelligence*, pp. 233-246, Budapest, Hungary, 2003

[10] H. Ahonen-Myka. Discovery of Frequent Word Sequences in Text. *Proc. of the ESF Exploratory Workshop on Pattern Detection and Discovery*, pp. 180-189, London, UK, 2002, Springer

[11] L. Aas, L. Eikvil. *Text Categorisation: A survey*. Raport NR 941, Norwegian Computing Center, 1999

[12] G. Salton, C. Buckley. Term Weighting Approaches in Automatic Text Retrieval. Information Processing and Management, **24**(5):513–523, 1998

[13] M. F. Porter. Snowball: A language for Stemming Algorithms, 2001 http://snowball.tartarus.org/texts/introduction.html

[14] V. Trón, P. Halácsy, P Rebrus, A. Rung, E. Simon, E, P. Vajda: morphdb.hu: magyar morfológiai nyelvtan és szótári adatbázis. *Proc. of MSZNY-05, III.. Magyar Számítógépes Nyelvészeti Konferencia*, pp. 169-179, Szeged, Hungary, 2005

[15] F. Sebastiani. Machine Learning in Automated Text Categorization. *ACM Computing Surveys*, **34**(1):1-47, 2002

[16] M. Meyer, C. Rensing, R. Steinmetz: Towards Using Wikipedia as a Substitute Corpus for Topic Detection and Metadata Generation in E-Learning. *Proc. of I2LOR, the 3rd Annual E-learning Conf. on Intelligent Interactive Learning Object Repositories*, November 2006

[17] E. Gabrilovich, S. Markovitch. Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis. *Proc. of IJCAI-07, the 20th Int. Joint Conf. on Artificial Intelligence*, pp. 1606-1611 Hyderabad, India, 2007

[18] M. Krier, F. Zaccà. Automatic Categorization Applications at the European Patent Office. *World Patent Information*, **24**:187-196, 2002