Reusing Knowledge Hidden in Wikipedia for Scalable Text Categorization

Marek Ciglan
Institute of Informatics
Slovak Academy of Sciences
Dúbravská cesta 9, Bratislava
marek.ciglan@savba.sk

Michal Laclavík
Institute of Informatics
Slovak Academy of Sciences
Dúbravská cesta 9, Bratislava
laclavik.ui@savba.sk

Alex Dorman
Magnetic Media Online
311 West 43rd St, Suite 1406
New York, NY 10036
alex@magnetic.com

ABSTRACT

In this paper we focus on the challenge of text classification into pre-defined hierarchy of categories, when no positive nor negative examples of textual content belonging to different categories are given. Our approach reuses the knowledge encoded in Wikipedia articles to build a classification model. Model construction relies on two steps; we first automatically assign a set of representative articles to each category; second, we use those seed articles to construct a dictionary of N-grams representing the given category. This approach allows us to define and work with a large number of categories, as long as we are able to represent category by Wikipedia articles. We have tested the approach on 1.9 million of Wikipedia articles and and evaluated it against 235 DBPedia categories assigned to these articles. We have successfully detected DBPedia categories with an F1 score of 59% . We have also compared our method with the standard LSI method on the same dataset.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]; H.3.1 [**Content Analysis and Indexing**]

Keywords

text categorization, Wikipedia, n-grams

1. INTRODUCTION

Text or document categorization is a well covered task in the scientific literature. The search for 'text categorization' in Google Scholar returns 40,000 scientific articles on this topic. Well established methods such as TF-IDF or LSI are used together with ML approaches like SVM [3]. State of the art approaches rely on model building from training sets with a large number of documents for each category.

In this paper we describe a categorization method, which requires only the categorization hierarchy and a corpus of Wikipedia to construct a categorization model. Our motivation is to provide a text classification approach that would categorize documents into a pre-defined taxonomy of categories and would be able to do that in the absence of a large training set for each category. Additionally, when using categorization at web scale, we need a simple and fast method, which is able to process thousands of documents per second. Our model reuses Wikipedia [4] as a source of human knowledge on a multitude of topics and categories. We exploit Wikipedia's textual content to build the categorization model. Traditional methods for text processing rely on a bag-of-words model and deal with single terms as a base unit. Intuitively, n-grams of words are important for categorization [1]. N-grams such as 'real estate' or 'human rights' represent particular categories very well, while being very generic when used as single keywords. In our work, we aim at exploiting n-grams for the text categorization task. Wikipedia is a useful resource for identifying n-grams, e.g. article names and their alternative names (redirect pages) explicitly define concise and meaningful n-grams.

Text classification has a multitude of possible applications. Our particular motivation came from the on-line advertising domain, more concretely from the domain of search retargeting, a form of targeted advertising where audiences are modeled based on the search queries users conduct on visited websites. By modeling user interests, search retargeting has the ability to find new customers, who never visited a marketer's website before. Search retargeting focuses on displaying advertisements to users who conducted searches for specific keywords or specific categories in the past. For this domain, categorizing queries as well as categorization of web pages where adds are or can be displayed, is the essential technique for user modeling and better user targeting. Companies focusing on search retargeting gather large number of user generated queries and visited websites that need to be categorized, on the order of thousands queries or websites per second. This challenge really needs a scalable and fast approach.

In the advertising domain, multiple taxonomies are used. Examples include the Google AdX taxonomy, containing more than 5,000 categories, or the IAB[1]taxonomy with almost 400 categories. In addition, each Ad Exchange provides their own taxonomy, which needs to be mapped, represented and detected well. Wikipedia can serve the purpose of mapping multiple taxonomies by representing taxonomy items using Wikipages.

[1]http://www.iab.net/QAGInitiative/overview/taxonomy

In this paper, we first describe how we model categories by n-grams computed from Wikipages, then we describe the categorization algorithm and we evaluate the approach on 1.9 million of Wikipedia articles with assigned DBPedia categories. We also compare our method with the standard LSI method on the same dataset. Finally we discuss and evaluate the performance of the algorithm. The algorithm achieves promising results with fast execution, which is based on an in-memory n-gram gazetteer matching approach.

2. MODELING CATEGORIES BY N-GRAMS

In this section, we first formulate the task, we then present our approach to the text classification with reuse of the Wikipedia data.

2.1 Problem statement

Given a predefined taxonomy of categories and a collection of documents, we want to classify the documents from the collection into the predefined categories. The categories are described only by short human readable strings (e.g. 'Sports/Basketball') and no training documents representing positive or negative examples of categories are given.

2.2 Modeling categories

As all the categories in the given taxonomy are represented only by short strings, containing a few terms, we first need a richer representation of categories. Our approach relies on reusing the human knowledge encoded in Wikipedia and use Wikipedia's rich textual content as a corpus to extend the representation of the categories. We model each category by exploiting its related Wikipedia articles. The first task is to identify a set of related Wikipedia articles for each category. We employ an information retrieval based approach; we model each Wikipedia article as a fielded document, and the collection of all Wikipedia articles is indexed. Wikipedia articles are modeled using multiple fields. We use a) article title, b) article text and c) titles of redirect pages pointing to the article as three fields. (In the experiments presented in this paper following weight where used for the three fields: a) 0.4 b) 0.25, c) 0.35.) Subsequently, we use each category as a query over the constructed index and retrieve top k matching documents (in the presented experiments $k = 200$). The retrieved documents are considered to be representative of the given category.

After the document retrieval step, we obtain the top k Wikipedia articles related to the string representation of a category. Instead of using the whole textual content of the retrieved articles, we only use their titles and scores from the information retrieval model to represent categories. The scores are scaled, so that the result at top 1 has the score of 1. To illustrate the model, we provide a simple example for the few first lines of a category labeled 'Airline':

```
Agent/Organisation/Company/Airline
        Airline                 1.0
        Airline timetable       0.905
        Bylina (airline)        0.857
        Airline deregulation    0.856
        ...
```

We subsequently extend generated n-grams by adding alternative names (AN) of the retrieved Wikipedia articles. Under the term AN, we understand the titles of redirect pages pointing to a Wikipedia article. For example, if there is Wikipedia article A and redirect pages $A1$ and $A2$ that redirect to article A; we consider titles of $A1$ and $A2$ to be ANs for article A. Returning to our previous example, the extended representation of 'Airline' category would be:

```
Agent/Organisation/Company/Airline
        Airline                 1.0
        Air carriers            1.0
        Commercial airline      1.0
        Commercial air transport 1.0
        ...
```

The dictionary of n-grams accompanied with the numerical score for each category in the given taxonomy is the model we use for the task of document classification.

2.3 Text classification procedure

We have developed a simple classification method that finds the occurrences of strings from the provided list of n-grams in an input text and sums the score for different categories. The categories with highest scores are considered to be the categories characteristic for the input text. More formally, each category c_i is represented by a set of n-gram keywords with scores. Each n-gram keyword k_m has an assigned score s_m.

$$C = \{c_1, c_2 ... c_k\}$$

$$c_i = \{(k_1, s_1), (k_2, s_2) ... (k_j, s_j)\}$$

When we categorize text, n-gram keywords are detected in the text. Set D contains all detected n-gram keywords in the categorized text with the number of occurrence n_m in the text.

$$D = \{(k_1, n_1), (k_2, n_2) ... (k_l, n_l)\}$$

Each n-gram keyword k_m can appear in more than one category.

$$s_{ci} = \sum_{D_j} n_m s_{mj} / \sum_{D} n_m s_m$$

Score s_{ci} of category c_i is computed as sum of all found keyword scores s_{mj} multiplied by the number of occurrences of this keyword n_m. The resulting score is normalized by the sum of all scores multiplied by the number of occurrences.

Implementation remarks. The approach for n-gram matching is a simple gazetteer based solution used in information extraction. We have used an implementation based on an in-memory char-tree, where all n-grams were loaded into memory creating an efficient character-tree structure [2]. This allowed us to have linear performance in n-gram detection in text, where we read text representing the Wikipedia article only once, firing detected n-grams with number of occurrences and categories represented by a score.

2.4 Example

Here we show how the categorization works on a concrete example of the first sentence representing the Wikipage of Bratislava.

Bratislava (formerly Slovak Prešporok; formerly Preßburg) is the capital of Slovakia and, with a population of about 460,000, the country's largest city.

We can see that three n-gram keywords 'country', 'the country' and 'city', were detected with scores computed from Wikipedia for related categories. Please note that 'the country' is the n-gram related to 'Village' category, because 'the country' is a redirect for 'rural area' Wikipage.

```
country
        Agent/Person/Politician/President:0.035888
        Place/PopulatedPlace/Country:1
the country
        Place/PopulatedPlace/Settlement/Village:0.02587
city
        Place/PopulatedPlace/Settlement/City:1
        Place/PopulatedPlace:0.046465
        Place/ArchitecturalStruct/Infrastructure:0.0747
        Place/PopulatedPlace/Settlement/Village:0.08665
        Place/PopulatedPlace/Settlement/Town:0.185952
        Agent/Person/Athlete/Cyclist:0.025997
        Place/.../RouteOfTransportation/Road:0.027028

Detected categories:
        Place/PopulatedPlace/Settlement/City:0.398637
        Place/PopulatedPlace/Country:0.398637
Categories assigned in DBPedia:
        Place, PopulatedPlace, Settlement

Precision: 60% Recall: 100%
```

Based on detected n-grams and the scores assigned to them, we have computed scores for each category. Please note that not all detected categories were returned. For example categories such as 'Cyclist' or 'Road' were discarded. We apply a rule, that when sorted by score, the next category is fired only if its relevance score s_{ci} is at least n% of the previous category score.

$$s_{cj} >= s_{ci}n$$

We have set up n experimentally to 40%. In this example, we have detected 3 relevant categories, but in fact all together 5 categories, if we take the whole category hierarchy into account. This is why Recall is 100% and Precision is 60% in this case. We have also detected the 'City' category, which is relevant but not assigned for the 'Bratislava' in DBPedia.

3. EXPERIMENT

In this chapter we describe the evaluation experiment, where we have categorized 1.9 million of Wikipedia articles into the DBPedia taxonomy. We have evaluated the n-gram based method described in this article as well as the standard LSI[5] method. First we describe how we represented taxonomy using the Wikipedia articles for both our and the standard LSI method and then we discuss the results. Detailed logs from the evaluation as well as per category results are available on-line[2].

We have used the DBPedia taxonomy[3] for text categorization. The DBPedia taxonomy contains 512 categories. We have used 235, because many of the categories are not assigned to any Wikipage. We have selected only categories with at least 100 Wikipages assigned to the category. The English Wikipedia contains of about 6 million of articles. DBPedia derives from Wikipedia and contains almost 2 million articles, where categories from the DBPedia taxonomy are assigned to these articles.

Table 1: Results on all Wikipedia articles with assigned DBPedia categories in %

	Recall	Prec	F1	at least one	no ctg
Sentence	61.08	57.76	**59.38**	69.11	22.88
S no AN	55.62	**59.50**	57.50	64.35	27.46
Abstract	71.27	49.07	58.12	81.56	9.24
A no AN	66.18	50.76	57.45	77.60	12.09
Text	**72.48**	43.61	54.45	84.45	5.60
T no AN	68.37	44.86	54.18	81.77	7.01
LSI 1	38.27	10.25	16.17	59.54	0
LSI 5	39.31	10.62	16.72	60.56	0
LSI 50	36.71	10.25	16.02	55.85	0

For the evaluation, we have used standard metrics of Precision, Recall and F1 measure, more concretely we have computed Micro Precision, Micro Recall and Micro F1, meaning that for each Wikipedia article we have computed these measures, as shown in section 2.4 and then we have computed averages of the measures. A bit problematic was to decide on how to define Precision in the case when no categories were returned by the evaluated methods. In the literature, sometimes it is defined as 0%, sometimes as 100%. We have decided to treat Recall as 0% but ignore Precision in such cases. This is why average Precision (Micro Precision) was computed only from results where the evaluated methods returned some results. For example, we have not returned any category for almost 23% of Wikipages when using only the first sentence for categorization, because no n-gram keywords were detected. Please see the last column named 'no ctg' in Table 3 to see the percentage of results where no categories were returned. The LSI method always returned some categories.

In the Table 3, we summarize the experiments, which covers the n-gram method applied on the first sentence ('S' or 'Sentence' in the table), on abstract ('A' or 'Abstract'), or whole text ('T' or 'Text') of Wikipedia articles. Rows with 'no AN' represent experiments where alternative names (Wikipage redirects) were not used. LSI experiments cover a different number of articles (1, 5, 50) to represent category, where text of articles was used to build the LSI model and represent categories.

Wikipedia categories were represented by n-grams derived from Wikipedia as described in section 2.2 for our method. For the standard LSI method, we needed a textual representation of categories. We have used a similar approach to retrieve the top k articles from Wikipedia based on category name, and we have used their texts to build the LSI model. The LSI model was built from 1.9 million Wikipedia articles using the existing library[5]. The building of the model took more than 1 day (see the computer configuration in 3.1).

[2]http://ikt.ui.sav.sk/research/TC/

[3]http://mappings.dbpedia.org/server/ontology/classes/

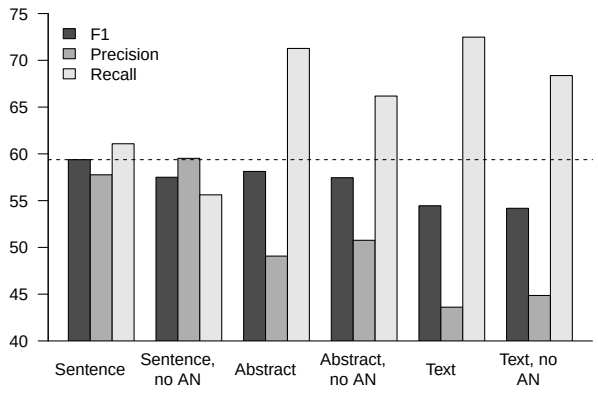

Figure 1: Results of the n-gram based categorization on Wikipedia articles, where whole text, abstract or only first sentence was used

In Table 3, we provide the results in the last three rows, where we have represented each Wikipedia category by the top 1, 5 or 50 Wikipedia articles retrieved from the same information retrieval model of Wikipedia as the one described in section 2.2. Then we have extracted the Wikipage texts and applied the LSI model on it. We can see that the best LSI Recall is around 39%, Precision about 10% and the best F1 measure is about 16% (see 'LSI' rows in Table 3). These poor results were achieved not because the LSI method is not suitable for this task, but because of poor representation of categories by the text of the retrieved documents. Our method achieved better results than standard LSI method, which is described in next paragraph.

We have categorized Wikipages based on our n-gram method using article text, article abstract and first sentence of the article for categorization. The best results were achieved on categorizing only the text of the first sentence of each article. See the Table 3 for the details as well as Figure 1. The column 'at least one' in Table 3 represents the success rate of returning at least one correct category for the results. This was best for our method when n-grams were matched in the whole text of an article (84%) but also quite high in the 'Sentence' case where it was above 69%. Achieved results are quite satisfactory, where the best F1 measure was more than 59% (the first line in Table 3 and the first column in Figure 1). This was achieved when using only the first sentence of an article for categorization. Recall was best when using whole text of articles (72%), but with a drop in Precision (almost 44%).

The main problem of categorizing based on first sentence is that our n-gram based method did not detected any category in 23% of articles, but based on the application needs, we can choose the right settings for the method. For example, the best Precision was achieved when using only the first sentence of an article and just titles of Wikipages as n-grams with no alternative names (second row in Table 3), but in this case we did not return any category for 27% of articles. On the other hand the Recall of our method is the best when categorizing whole text of the article also using alternative names as n-grams. In the future, we would like to improve our method by extending n-grams with a wider set and better scoring based on using both texts and structure (e.g. links or section names) of Wikipedia.

3.1 Performance and Scalability

We have tested the performance of developed method. We have run the text categorization experiment on whole text of Wikipedia articles (it is much faster when categorizing only using the first sentence or abstract). Evaluation was done on a computer with the following configuration: Intel(R) Xeon(R) CPU E5-2620 2.00GHz; 2x6 core processors; 32GB RAM. We were able to categorize 1.9 million articles within 50 minutes and 14 seconds, meaning that we are able to categorize 632 documents within a second in single thread. The method is fast because it is matching n-gram keywords in text based on an in-memory gazetteer [2] with near linear complexity. The process of categorization is quite fast, it can also run independently on separate threads and thus benefit from multi-core machines while reusing the same in-memory tree structure of a list of n-gram keywords. In this case we have consumed 2.3GB of memory for loading about 170 thousand of n-gram keywords. Memory can be also used more efficiently, for the details, please see[2].

4. CONCLUSIONS

In this paper, we have shown how human knowledge hidden in Wikipedia can be used for fast and efficient text categorization using various taxonomies, where the only requirement is to have categories represented by a meaningful name. We have evaluated the approach on the DBPedia taxonomy and 1.9 million Wikipages, where we have achieved an F1 score of 59%. In addition we have compared it with the standard LSI categorization method, within same principles having no training set for the given taxonomy. We believe the results can be further improved by better n-gram representation of categories for the given taxonomy. So far we have tested the method only on Wikipedia, where the n-grams representing categories were extracted. We believe it can be used for the general task of web categorization within rich taxonomies in the context of on-line advertising or other domains, which will be the focus of our future work.

5. ACKNOWLEDGMENTS

This work is supported by Magnetic, Inc. and by VEGA 2/0185/13, VENIS FP7-284984, CLAN APVV-0809-11 and ITMS: 26240220072 projects.

6. REFERENCES

[1] W. B. Cavnar, J. M. Trenkle, et al. N-gram-based text categorization. *Ann Arbor MI*, 48113(2):161–175, 1994.

[2] S. Dlugolinsky, G. Nguyen, M. Laclavik, and M. Seleng. Character gazetteer for named entity recognition with linear matching complexity. In *Proceedings of WICT*, WICT'13, pages 364–368. IEEE, 2013.

[3] T. Joachims. *Text categorization with support vector machines: Learning with many relevant features.* Springer, 1998.

[4] O. Medelyan, D. Milne, C. Legg, and I. H. Witten. Mining meaning from wikipedia. *Int. J. Hum.-Comput. Stud.*, 67(9):716–754, Sept. 2009.

[5] R. Rehurek. Subspace tracking for latent semantic analysis. In *Proceedings of ECIR*, ECIR'11, pages 289–300, Berlin, Heidelberg, 2011. Springer-Verlag.