WEB PAGE MULTI-LABEL CLASSIFICATION FOR FILTERING CONTENT FROM THE WEB

Juraj Hresko
Faculty of Informatics, Masaryk University
Botanicka 68, Brno, Czech Republic
e-mail: juraj.hresko@mail.muni.cz

Abstract

In this paper, we describe a simple approach to filter unwanted web pages, according to their content. The result of this work is a demo of an application that is usable in real-time filtering and in non-real-time indexing of any given web pages. We describe a proposed technique step by step, while discussing possible alternative ways for each part. In the end we discuss the overall quality and proposed next steps that could lead to a fully usable business application.

Keywords: *multi-label classification, web mining, machine learning, content filtering*

1. INTRODUCTION

The World Wide Web is unexceptionably the largest database that mankind has created. The amount of data resulted in many new problems. The data has to be organized in order to ease manual browsing. It is also desirable to have the tools which could search the Internet for user inputted queries. For those two types there are two appropriate approaches.

The latter approach is far creating catalogs of web pages. This was the favorable approach in the beginning of the web. The catalogs were built manually at first, but later the methods were designed to automatize this process. On the other hand, man created catalogs can be found nowadays still, e.g. the largest one www.dmoz.org that contains over five million pages separate in over a million categories as we can read on the home page of Open Directory Project.

The second approach is full text searching. It started as normal word matching, but now it is composed of many specialized components that perform reformulation of query, identifying the meaning of it and so on.

Besides categorizing and searching there is another task that includes knowledge of the document content – web page filtering. It can be used in many different ways, including filtering mature content for children, inappropriate content in offices or even blocking uncomfortable pages in totalitarian states.

The problem with filtering web pages can be handled as a classification task, which is well covered by current flourishing research in area of web mining that uses methods for machine learning. The classification handles the problem of selecting a proper label (from at least two) for an examined example (web document in our case).

Our case consists of several problems. First, it is not a standard classification task, but a multi-label classification. This means that there can be more than just one class for every given document. Second, we have to adjust the application so it can handle tens of sparsely distributed classes. The third is that it has to be suitable for real-time as well as for non-real-time indexing.

The main contribution of this paper is that it provides very simple and language independent approach which is usable for multi-label classification problem with very large set of classes. These parameters implies potential for more precise description of the content of a web pages.

2. RELATED WORKS

Labeling the web pages is a problem that is discussed in many works e.g. [2][3][5]. As a part of the process it was necessary to find a proper method for selecting attributes which would fit well for text classification. This part the of process was covered by articles [4][1]. For our purpose, it was also needed to take into account works that paid interest in multi-label classification [8][6].

3. EXPERIMENTAL SETUP

As we mentioned earlier our task involves multiple classes that are possible to fit each page. These vary from common ones like *News and Magazines*, *Web Based E-mails* or *Porn* to specific ones as *Sects, Illegal Drugs* and *Insurance*. Total number of categories used was 61.

Our experimental set consists of 80,000 labeled URLs, which were picked at random from a bigger database for this purpose. Because of this, the distribution of all classes was similar to real.

Every page could be labeled as one to three classes. The distribution is summarized in Table 1.

Table 1. Distribution of labels per page in experimental data

Categories for page	Occurrence in experimental data
0	0.41 %
1	64.45 %
2	31.75 %
3	3.38 %

As for the language, most of pages in the experimental set were written in Czech, with fractional part in Slovak and English.

4. PROPOSED METHOD

The whole process of creating this classifier consists of few obligatory steps. In the beginning we needed to download the page, then create a model for it and then use the model for learning the classifier or to label the page according to the already built classifier.

4.1. Model of document

First we needed to decide how to get the content of the page and how to represent each one. The following phase was to mine knowledge from the created database of labeled samples. Finally the resulted classifier had to be evaluated to find out if the proposed method was worthwhile for implementing real usage.

In the beginning we had to choose the type of representation of each sample. We considered each page as a bag of words and we picked a traditional vector space model that was the suitable one for this task. It must be mentioned that this kind of model can't be used to process web pages that contain only a few words or no words at all. It includes e.g. the pages built solely with Adobe Flash technology. In this case, the program should get an appropriate response.

As for preprocessing, it would only consist of basic inevitable steps in order to keep the overall time performance of algorithm high. After downloading and encoding recognition, the content of the page was encoded in UTF-8. In the next step we removed all tags and scripts with one exception. It is reasonable to keep the information about text structured tags like headings and titles. We acquired a vertical file with a column for words (which were transposed to lowercase) and a column for structure tags. The terms were then weighted by a number of occurrences, with a different weight for each occurrence in a standard paragraph and in titles or headings. The weights for each part of the page structure are shown in Table 2. As a result we got a vector where each dimension represents one word from the document and the value was determined by Structure-oriented Weighting Technique (SWT) [4].

Table 2. Weights for terms used in structured parts

Structure tag	Weight
title	10
h1	5
h2	3
h3	2
none	1

This method is proposed in [4] and is defined by the function showed in (1). «Where e_k is a structure tag, $w_k(e_k)$ represents the function described in Table 2. and $TF(t_i,e_k,d_j)$ denotes how many times term t_i is present in the element e_k of the HTML document d_j»[4].

$$SWT_w(t_i,d_j)=\sum_{e_k}\left(w(e_k)\cdot TF(t_i,e_k,d_j)\right) \tag{1}$$

It is necessary to point out that we didn't remove any stop words or provide stemming. The reason why we did so was simple. The stop words would be eliminated in the next step — when the attributes would be chosen. As for stemming — it could be a time consuming process (for Slavic languages). The other possibility to reduce word number could be by using a utility for morphological analysis, but for a business purpose it had to be bought or created, which wasn't suitable as we were only examining this task for proposing a simple solution.

Table 3. Processing of data from page

HTML	Vertical		Model	
<title>Interesting article</title> <h1>The article<h1> This is the main part of article. <h1>The end<h1> This is the end of our article.	interesting	title		
	article	title		
	the	h1		
	article	h1		
	this	none		
	is	none	article	17
	the	none	the	12
	main	none	interesting	10
	part	none	end	6
	of	none	this	2
	article	none	is	2
	the	h1	of	2
	end	h1	main	1
	this	none	part	1
	is	none	our	1
	the	none		
	end	none		
	of	none		
	our	none		
	article	none		

4.2. Attribute selection

The next step in building the classifier was to select proper attributes. While we decided to use the set of classifiers (a unique one for each class), we needed a distinct set of attributes for each.

For this part of the process it was necessary to build a dictionary from training samples by joining all the word verticals we got from the last step. To reduce the amount of data in the dictionary, we had to cut off words that had a frequency lower than five. Then we used information gain (*IG*) for every class to rank the terms (Equation (2). - «where *Ex* is the set of training examples, *a*∈*Attr* is an attribute *value*(*x*,*a*) with *x*∈*Ex* defines the value of attribute *a* for example *x* and *H* stands for entropy» [9]).

$$IG(Ex,a) = H(Ex) - \sum_{v\in(values(a))} \left(\frac{\left| \{x \in Ex \mid value(x,a) = v\} \right|}{\left| Ex \right|} \right) \cdot \\ \cdot H(\{x \in Ex \mid value(x,a) = v\}) \qquad (2)$$

As a result of this ranking procedure we gained 61 different word lists, each containing the same number (tens of thousands) of terms. While trying to avoid overfitting the classifiers we had to take into account the number of samples for each class. The amount of positive examples for each of the classes vary from a few tens to a few thousands, while the arithmetic average was around 2,000. So considered this fact we chose the initial limit for the number of attributes — two thousand. In later experiments we found out that by using a few thousand would be enough for the real deployment also.

4.3. Machine learning method

As in the standard classifier building procedure, we had to choose a suitable learning algorithm for this task. We decided to include: Multinomial Naïve Bayes (MNB), Support Vector Machines (SVM) and Random Forest (RF)[7].

For every classifier we used its implementation from Weka Data-mining Software. SVM was used with linear kernel. Random Forest was set up with parameters — 50 trees in all and 50 random attributes considered in each node.

As initial training data, we took a set of positive examples for a specific category which seemed average and we picked the same amount of negative examples at random. Then we used ten-fold cross-validation to evaluate the methods.

Table 4. Results for classifiers for testing category

Result	MNB	SVM	RF
Precision	84.3 %	84.3 %	**88.8 %**
Recall	84.3 %	84.3 %	**87.5 %**
F-measure	84.2 %	84.2 %	**87.3 %**

According to the test results we considered the Random Forest as being favorable try. Besides the results, the other pros were readability of derived knowledge, overfitting robustness of that algorithm and lower memory consumption during learning process.

While opposing the multi-labeling classification problem, we decided to choose the most straightforward method that uses the set of $|C|$ binary classifiers where C is the set of all classes [6].

With algorithm chosen we trained classifiers with almost the same setup for each class. Because of requested feature of developed application which was an emphasis on minimizing the number of False Positive errors, we had to make one change in setup. We had to change the weights for negative samples in training set to 5, by using meta classifier which allowed to set the weights for each type of error.

5. EVALUATION

Trained classifiers were evaluated by ten-fold cross-validation. Their overall quality was heavily dependent on two factors. The first was the number of used positive examples because as we noticed earlier there were quite imbalanced classes (such as pages about sects or sexual health). The other factor was ambiguity of some classes. For example the category *Business* had to involve web presentations of companies with any subject on business and *Social networks* category pages could contain almost any possible textual content as well. While manually controlling the results we observed that the main cause of false positive classifications were represented with quite long lists of proposed classes for the sample. According to the knowledge that each page could have a limited amount of categories, we decided to sort the proposed classes by their probability and took the first two of them. Then we used two measurements to compare the quality of classifiers for each category — precision and recall.

The average classification precision for each classifier was 81.78% when differing from only a few very poor, such as 2 — 30 % to a few exceptional classifiers with correct classification for over 90%. The average recall observed was 54,4% ranging from 25 — 70% for almost all classes.

Table 5. Confusion matrix for unoptimized classification[1]

Correct classification	Proposed classification	
	+	-
+	32,199	7,174
-	26,994	2,531,228[2]

[1] Sum of confusion matrices for each classifier in the classifiers set.

[2] High number of True Negative classifications is caused by number of possible categories. For every example from cca 40,000 are most of categories wrong.

In the last step we had to propose how the created classifiers should be used in a real service. We wanted to propose one solution with maximized precision. For maximizing the precision, we took ¼ of experimental data as the training data and found the optimal threshold for each classifier, so it wont label a sample unless it is sure enough [2]. While the result of Random Forest classifier was a probabilistic value with two decimal points precision, we just moved through the interval until we got precision near one hundred percent on training data. With those values used as a threshold and omitting the least presented ten classes (which classifiers were unusable because of the number of training examples) we managed to get an overall precision of 96.31% with a recall of 31.7%. There were twelve categories which performed best — with ballanced values of precision (over 90%) and recall (over 40%).

6. CONCLUSIONS

While we took into account the fact that the at least a very small part of data that we used for training and testing was outdated (pages could be canceled or changed), the results looked quite promising. The problem with some categories specifically *Social Networks* could be resolved easily by replacing the classifier with a list of selected domains. The survey of results and manual testing showed that the approach can provide useful information about examined documents, because of the used combinations of categories. For example it was able to recognize pages from Czech government like *Government* category and besides that it recognized if the article showed discussed environmental issues or financial ones (classes *Environment* and *Money/Finance*). In the same way it not only discovered that the page is a blog, but it could specify if it is a personal or e.g. about hacking. Other findings were that it works well on mostly banned pages that contained porn and violence. This is very useful, because standard blacklists for those kinds of pages got outdated in a time and they could not cover all personal pages with adult content, which seems to represent not such a big problem for our approach.

The next improvements could be made by using the hypertext links from the page. Although this will not be applicable for real-time processing, because it would cripple the time performance. Other possibilities includes further experiments with tuning the classifiers one by one, with choosing different parameters for learning the classification rules.

REFERENCES

1. **Yang, Y. and Pedersen, J.A** Comparative Study on Feature Selection in Text Categorization. ICML 1997: 412-420
2. **He, X., Duan, L., Zhou, Y., Dom, B.**: Threshold selection for web-page classification with highly skewed class distribution. WWW 2009: 1081-1082

3. **Tsukada, M., Washio, T. and Motoda, H.** 2001. Automatic Web-Page Classification by Using Machine Learning Methods. In Proceedings of the First Asia-Pacific Conference on Web Intelligence: Research and Development (WI ‘01), Ning Zhong, Yi Yu Yao, Jiming Liu, and Setsuo Ohsuga (Eds.). Springer-Verlag, London, UK, UK, 303-313.
4. **Riboni, D.**, «Feature Selection for Web Page Classification». In EURASIA-ICT 2002 Proceedings of the Workshop, pp. 473-478. Editor: Tjoa M., Austrian Computer Society. PDF
5. **Fiol-roig, G., Miró-julià, M. and Herraiz, E.**, 2011. Data Mining Techniques for Web Page Classification. Highlights in Practical Applications of Agents and Multiagent Systems, 2(2), p.61-68
6. **Tsoumakas, G. and Katakis, I.**, 2007. Multi-label classification: An overview.International Journal of Data Warehousing and Mining, 3(3), p.1–13.
7. **Breiman, L.** 2001. Random forests. Available at ftp://ftp.stat.berkeley.edu/pub/users/breiman/randomforest2001.pdf
8. **Read, J., Pfahringer, B., Holmes, G. & Frank, E.**, 2009. Classifier chains for multi-label classification. In Proceedings of European conference on Machine Learning and Knowledge Discovery in Databases 2009, Part II, LNAI 5782, p. 254-269
9. http://en.wikipedia.org/wiki/Information_gain_ratio