Automatic Content-based Categorization of Wikipedia Articles

Zeno Gantner
University of Hildesheim
Machine Learning Lab
gantner@ismll.de

Lars Schmidt-Thieme
University of Hildesheim
Machine Learning Lab
schmidt-thieme@ismll.de

Abstract

Wikipedia's article contents and its category hierarchy are widely used to produce semantic resources which improve performance on tasks like text classification and keyword extraction. The reverse – using text classification methods for predicting the categories of Wikipedia articles – has attracted less attention so far. We propose to "return the favor" and use text classifiers to improve Wikipedia. This could support the emergence of a virtuous circle between the *wisdom of the crowds* and machine learning/NLP methods.

We define the categorization of Wikipedia articles as a multi-label classification task, describe two solutions to the task, and perform experiments that show that our approach is feasible despite the high number of labels.

1 Introduction

Wikipedia's article contents and its category hierarchy are widely used to produce semantic resources which improve performance on tasks like text classification and keyword extraction (Banerjee, 2007; Gabrilovich and Markovitch, 2007; Minier et al., 2007; Mihalcea and Csomai, 2007; Wang and Domeniconi, 2008; Medelyan et al., 2008). The reverse – using text classification methods to improve Wikipedia's article-category mappings – has attracted less attention (Fu et al., 2007).

A system that automatically suggests categories for Wikipedia articles will help to improve the encyclopedia for its users and authors, as well as the semantic resources created from it.

The complexity of Wikipedia's category systems[1] and sheer number of categories make it hard for – possibly inexperienced – authors to assign categories to new or existing articles. As of February 2009, the German Wikipedia has about 886,000 articles, which belong to about 64,000 categories. For the English Wikipedia, those numbers are even higher.[2]

Classical document classification data sets like Reuters RCV1-V2 (Lewis et al., 2004) have around 100 different categories. In comparison, the automatic categorization of Wikipedia articles is a challenging task, as it involves tens to hundreds of thousand categories. For such large-scale classification problems, particular attention is necessary to deal with both training and prediction complexity, as well as imbalanced class distributions.

In this article, we present the problem of content-based article categorization in Wikipedia, and suggest an evaluation protocol as well as two content-based methods for solving this problem.

2 Problem Statement

Let $X \subseteq \mathcal{X}$ be the set of all articles and L be the set of all category labels in one of Wikipedia's language versions. Each article $x \in X$ is assigned a set of $k(x)$ category labels $\{l_1, \ldots, l_{k(x)}\} \subseteq L$.

In this context, one can think of several prediction problems: Given an article x without category information, predict all the article's categories. This scenario is typical for newly created articles, thus we call it the **new article problem**. Another prediction task would be to predict the missing categories for an article with existing, but incomplete category information (**missing categories problem**). Such a condition can occur e.g. if a new category is created and the creator of the new category does not include all existing articles that should be assigned to that category. In this pa-

[1]We use the plural here, as each language version has its own category hierarchy. The categories may be linked across languages using so-called interlanguage links.

[2]http://stats.wikimedia.org/

		$f_i(x)$ 1	-1
$\hat{f}_i(x)$	1	tp_i	fp_i
	-1	fn_i	tn_i

Table 1: Confusion matrix for class i.

per, we will concentrate on the *new article problem*.

Such a problem is a so-called *multi-label*, or *any-of* classification task, as opposed to *single-label* (*one-of*) classification (Manning et al., 2008). Multi-label classification can be expressed as a set of binary classification problems:

$$f(x) = \{l_i | f_i(x) = 1\}, \tag{1}$$

where $f_i : \mathcal{X} \to \{-1, 1\}, 1 \leq i \leq |L|$ are indicator functions for class l_i, i.e. $f_i(x) = 1$ iff. article x is annotated with the category label l_i.

The associated learning problem is to find a prediction model $\hat{f}$ that predicts categories for given articles as good as possible, according to a given loss function.

We choose micro- and macro-averaged F_1 as loss functions. Micro-averaged F_1 is computed from the complete confusion matrix, while macro-averaged F_1 is the average F_1 computed from class-wise confusion matrices. Micro-averaged measures tend to measure the effectiveness of a classifier on the large categories, while macro-averaging gives more weight to smaller categories (Manning et al., 2008).

$$\text{F}_1^{\text{macro}} := \frac{1}{|L|} \sum_{i=1}^{|L|} \frac{2 \cdot \text{tp}_i}{2 \cdot \text{tp}_i + \text{fp}_i + \text{fn}_i}, \tag{2}$$

where tp_i is the number of true positives, fp_i the number of false positives, and fn_i the number of false negatives for class i (see Table 1).

$$\text{F}_1^{\text{micro}} := \frac{2 \cdot \text{tp}}{2 \cdot \text{tp} + \text{fp} + \text{fn}}, \tag{3}$$

where $\text{tp} = \sum_{i=1}^{|L|} \text{tp}_i$ is the overall number of true positives, $\text{fp} = \sum_{i=1}^{|L|} \text{fp}_i$ the overall number of false positives, and $\text{fn} = \sum_{i=1}^{|L|} \text{fn}_i$ the overall number of false negatives.

F_1 is widely used in information retrieval and supervised learning tasks. While providing a balance between precision and recall, optimizing for F_1 "forces" the prediction method and the respective learning algorithm to decide which category labels to predict and which ones not – just predicting a ranking of labels is not sufficient. This is motivated by the intended use of the prediction method in a category suggestion system for Wikipedia articles: Such a system cannot present an arbitrarily high number of (possibly ranked) suggestions to the user, who would be overwhelmed by the amount of information. On the other hand, if there is a fixed low number of suggestions, there would be the danger of correct category labels being left out.

3 Methods

There are many multi-label classification models in the literature, which are either adaptions of existing single-label models, or models generated by transformation of the multi-label problem to single-label problems, which are then solved using again existing single-label models. Tsoumakas et al. (2009) give an overview of multi-label classification methods.

Wikipedia articles are hypertext pages. For classifying hypertext pages, there are two obvious kinds of features: (i), there are *content-based features*, like words or n-grams contained in the articles, and (ii), there are *link-based features*, such as in- and outgoing article links, links to external web pages, and the (estimated or actually known) categories of the linked articles. Past research on relational learning and hypertext classification (Lu and Getoor, 2003) has shown that both kinds of features are useful, and that the strongest methods combine both. It makes sense to investigate content-based features as well as link-based features, because improvements in any of the two can lead to overall improvements. The work presented here focuses on content-based features.

A naive approach would be to directly take the binary representation of multi-label classification (equation 1), and then to train binary classifier models like support-vector machines (SVM, Cortes and Vapnik (1995)):

$$\hat{f}_{\text{naive}}(x) := \{l_i | \hat{f}_i(x) = 1\} \tag{4}$$

As the training of a traditional binary SVM classifier does not optimize towards the given multi-label loss function, but for accuracy, we do not expect the best results from this method.

If we want better multi-label predictions, changing the threshold of the binary decision functions is a straightforward solution. We employed two well-known thresholding strategies, ranking cut (RCut) and score cut (SCut, Yang (2001)), to predict Wikipedia categories.

RCut sorts all labels according to their binary prediction score $\hat{f}_i^*$, and selects the t top labels:

$$\hat{f}_{\text{rcut}}(x) := \text{argmax}^t_{1 \leq i \leq |L|} \hat{f}_i^*(x), \qquad (5)$$

where $\text{argmax}^t_{a \in A} g(a)$ refers to the t elements of A with highest value $g(a)$. The value of the hyper-parameter threshold t can be chosen empirically on a hold-out set.

SCut uses an individual decision threshold s_i for each label:

$$\hat{f}_{\text{scut}}(x) := \{l_i | \hat{f}_i^*(x) \geq s_i\} \qquad (6)$$

Good threshold values s_i can be determined during training. Algorithm 1 shows a category-wise optimization of the threshold values as described by Yang (2001). Because it tunes the threshold s_i for each category based on the F_1 measure over that category, it optimizes for macro-averaged F_1. If we are able to find optimal thresholds for each category, then we will achieve optimal macro-F_1 performance, as the following lemma says.

Lemma 1 *Let*

$$s_i := \mathit{argmax}_{s \in S} F_1(X, Y_i, \hat{f}_i), \qquad (7)$$

$$\hat{f}_i(x) := \begin{cases} 1, & \mathit{if}\ \hat{f}_i^*(x) > s \\ -1, & \mathit{otherwise} \end{cases} \qquad (8)$$

Then

$$(s_1, ..., s_{|L|}) = \mathit{argmax}_{(s'_1, ..., s'_{|L|})} F_1^{\mathit{macro}}(X, Y, \hat{f}), \qquad (9)$$

$$\hat{f}(x) := \{l_i | \hat{f}_i^*(x) > s'_i\}) \qquad (10)$$

i.e., the component-wise binary F_1 optimization yields the F_1^{macro}-optimal multi-label threshold.

Proof: The components of the sum in the definition of macro-averaged F_1 (Equation 2) are exactly the class-wise F_1 values. The choice of s_i influences only the part of the sum $\frac{2 \cdot \text{tp}_i}{2 \cdot \text{tp}_i + \text{fp}_i + \text{fn}_i}$ belonging to i. Thus each s_i can be optimized independently.

Representing each category label as binary prediction problem, as in the work presented here, requires $|L|$ binary classifiers. There also exist methods that use $|L|^2$ binary classifiers (Mencia and Fürnkranz, 2008), which is not feasible if L is large.

Algorithm 1 Macro-averaged F_1 optimization for SCut

```
Input: binary classifiers $(\hat{f}_i^*)$, $\hat{f}_i^* : \mathcal{X} \to S$; training instances $X \subseteq \mathcal{X}$ and labels $Y \in \mathcal{P}(L)^{|X|}$
Output: thresholds $(s_i)$
1: for $i = 1$ to $|L|$ do
2:     $Y_i \leftarrow$ binary labels for category $i$ generated from $Y$
3:     $s_i \leftarrow \text{argmax}_{s \in S}$ $\text{F}_1$-measure for $\hat{f}_i^*$ with threshold $s$ on $X, Y_i$
4: end for
5: return $(s_i)$
```

4 Experiments

To demonstrate the general feasibility of the automatic categorization of Wikipedia articles, we conducted experiments on a subset of the German Wikipedia. In this section, we describe the extracted data sets, the evaluation protocol, and discuss the results.

4.1 Category Data

To generate the data set for the experiment, we used the official database dumps of the German Wikipedia, generated December 6, 2008.[3] We then extracted all articles belonging to the category *Eishockey* ("ice-hockey") or to one of its descendants, and removed all category labels from outside the chosen category sub-graph, and all category labels of categories containing less than 5 articles. We proceeded identically for the category *Philosoph* ("philosopher").

Feature generation was performed as follows: First, we removed all wiki markup from the article source code. Second, we used Mallet (McCallum, 2002) to generate bag-of-words representations of the articles. All tokens were converted to lower case, and tokens occurring in only one article were removed. We conducted no stopword removal, nor stemming. Finally, we normalized the feature vectors to sum up to one.

Table 2 shows some properties of the data. $|X|$ is the number of instances, $|L|$ the number of distinct category labels; the fourth column contains the number of features (words) in the data set.[4]

[3] `http://download.wikimedia.org`

[4] The data can be downloaded from `http://www.domain/path`.

top category	$\lvert X\rvert$	$\lvert L\rvert$	# features
Philosoph	2,445	55	68,541
Eishockey	5,037	159	36,473

Table 2: Data set properties.

4.2 Evaluation Protocol

Train-Test Split

For the experiment, we randomly separated the data sets into 80% of the articles for training, and 20% for testing. To evaluate the *new article problem*, we removed all category labels from the articles in the test sets.

Training

As an experimental baseline, we used a static classifier (*most-frequent*) that always predicts the most frequent categories, regardless of the article.

We implemented the RCut and SCut strategies using linear support-vector machines from the LIBSVM library (Chang and Lin, 2001) for the underlying binary classification task. For each category, we used 5-fold cross-validation to find a good value for the hyperparameter C (Hsu et al., 2003). As SVMs perform only binary decisions, but do not yield scores suitable for ranking the labels, we used LIBSVM's modified version of Platt's method (Platt, 2000) to obtain probabilities, which are used as scores for the RCut rankings and the SCut decisions. As SCut's threshold search goes over an infinite set $S = [0, 1]$ (Algorithm 1, line 3), we did an approximate search over this interval with step size 0.01. For RCut and *most-frequent*, we report results for all thresholds $1, \ldots, |L|$. In an application setting, we would have to determine a suitable t using a hold-out data set.

4.3 Results and Discussion

The results can be seen in Table 3 and Figure 1 and 2. Both methods clearly perform better than the baseline. For macro-averaged F_1 on *Eishockey*, SCut performs better than RCut, which is not surprising, as this method is optimized towards macro-averaged F_1. For *Philosoph*, RCut with a rank threshold of $t = 3$ has a little bit (by 0.005) higher macro-averaged F_1 result, but this is likely not a significant difference.

The experiments show that simple models like the transformation from multi-label to binary problems, combined with thresholding strategies like SCut and RCut, are suitable for the categorization of Wikipedia articles: The methods achieve a good prediction quality, while the number of underlying binary classifiers scales linearly (see Section 3).

5 Conclusion and Future Work

In this article, we view the categorization of Wikipedia articles as a multi-label classification problem and report experiments on a subset of the German Wikipedia. The experiments show that there are suitable models for the categorization of Wikipedia articles.

We propose to use machine learning algorithms in order to improve the category assignments of Wikipedia articles. While data from Wikipedia is already widely used to improve text classification systems, it may be desirable to "return the favor" and use text classifiers to improve Wikipedia. This could support the emergence of a virtuous circle between the wisdom of the crowds and machine "intelligence", i.e. machine learning and NLP methods.

Wikipedia category data could be used as well for generating publicly available, large-scale (hierarchical) multi-label classification benchmark collections with different characteristics. Furthermore, it could provide the basis for multilingual document classification data sets.

To be able to provide category suggestions for large Wikipedias like the German, the Spanish or the English one, we will extend our experiments to larger subsets, and finally to all of the German and English Wikipedia. In order to achieve this, we will also investigate hierarchical multi-label classification methods (Liu et al., 2005; Cai and Hofmann, 2004; Cesa-Bianchi et al., 2006) and faster training algorithms for linear SVMs and logistic regression (Fan et al., 2008; Shalev-Shwartz et al., 2007). Given that we use $|L|$ binary classifiers for our models, this should be feasible, even for large numbers of categories. It would also be interesting to compare our methods to the work by Fu et al. (2007), which concentrates on link-based categorization of Wikipedia articles.

Other promising research directions are the examination of Wikipedia-specific features, and the survey of large-scale multi-label classification algorithms that take into account dependencies between labels.

	micro-averaged			macro-averaged		
	P	R	F_1	P	R	F_1
method	*Philosoph*					
most-frequent ($t = 1$)	0.489	0.315	0.383	0.009	0.019	0.012
most-frequent ($t = 55$)	0.028	1.0	0.055	0.028	1.0	0.049
RCut ($t = 2$)	0.522	0.674	0.589	0.252	0.283	0.244
RCut ($t = 3$)	0.395	0.764	0.520	0.240	0.379	0.266
SCut	0.341	0.735	0.466	0.225	0.350	0.261
method	*Eishockey*					
most-frequent ($t = 2$)	0.214	0.162	0.185	0.001	0.007	0.003
most-frequent ($t = 159$)	0.008	1.0	0.016	0.008	1.0	0.017
RCut ($t = 1$)	0.829	0.628	0.715	0.499	0.472	0.494
RCut ($t = 2$)	0.526	0.796	0.633	0.406	0.599	0.497
SCut	0.646	0.806	0.717	0.461	0.630	0.554

Table 3: Results for data sets *Philosoph* and *Eishockey*.

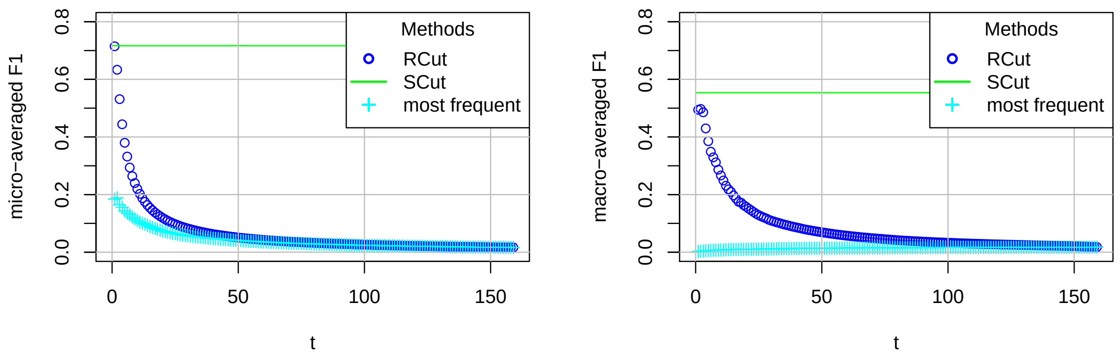

Figure 1: Method comparison for F_1 on data set *Eishockey*. SCut does not depend on t.

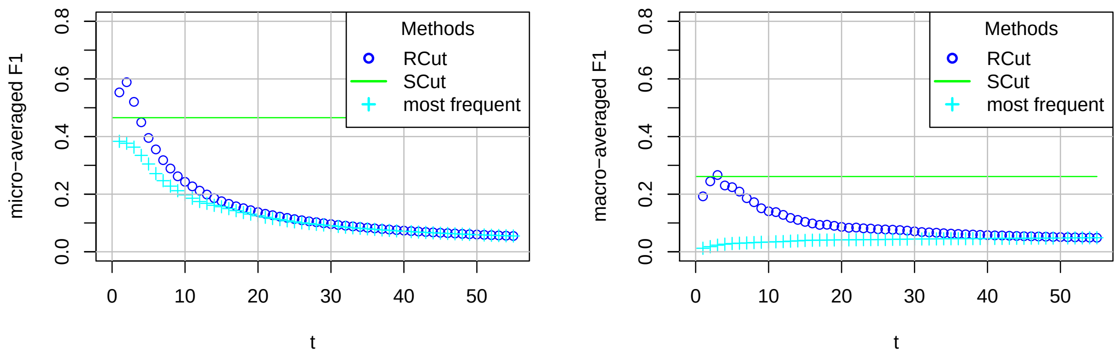

Figure 2: Method comparison for F1 on data set *Philosoph*. SCut does not depend on t.

Acknowledgments

The authors gratefully acknowledge the partial co-funding of their work through the European Commission FP7 project MyMedia (www.mymediaproject.org) under the grant agreement no. 215006.

References

Somnath Banerjee. 2007. Boosting inductive transfer for text classification using Wikipedia. In *ICMLA '07: Proceedings of the Sixth International Conference on Machine Learning and Applications*, Washington, DC, USA. IEEE Computer Society.

Lijuan Cai and Thomas Hofmann. 2004. Hierarchical document categorization with support vector machines. In *Proceedings of the 13th ACM International Conference on Information and Knowledge Management (CIKM '04), November 8-13, 2004, Washington, D.C., USA*. ACM Press, New York, NY, USA.

Nicol Cesa-Bianchi, Claudio Gentile, and Luca Zaniboni. 2006. Incremental algorithms for hierarchical classification. *Journal of Machine Learning Research*.

Chih-Chung Chang and Chih-Jen Lin, 2001. *LIBSVM: a library for support vector machines*. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Corinna Cortes and Vladimir Vapnik. 1995. Support–vector networks. *Machine Learning*, 20.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9.

Linyun Fu, Haofen Wang, Haiping Zhu, Huajie Zhang, Yang Wang, and Yong Yu. 2007. Making more Wikipedians: Facilitating semantics reuse for wikipedia authoring. In *ISWC/ASWC 2007*.

Evgeniy Gabrilovich and Shaul Markovitch. 2007. Harnessing the expertise of 70,000 human editors: Knowledge-based feature generation for text categorization. *Journal of Machine Learning Research*.

Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. 2003. A practical guide to support vector classification. Technical report, Department of Computer Science, National Taiwan University.

David D. Lewis, Yiming Yang, Tony G. Rose, G. Dietterich, Fan Li, and Fan Li. 2004. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*.

Tie-Yan Liu, Yiming Yang, Hao Wan, Hua-Jun Zeng, Zheng Chen, and Wei-Ying Ma. 2005. Support vector machines classification with a very large-scale taxonomy. *SIGKDD Explorations*, (1).

Qing Lu and Lise Getoor. 2003. Link-based classification using labeled and unlabeled data. In *ICML Workshop on "The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining"*.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York.

Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu.

Olena Medelyan, Ian H. Witten, and David Milne. 2008. Topic indexing with Wikipedia. In *Proceedings of the Wikipedia and AI workshop at AAAI-08*. AAAI.

Eneldo Loza Mencia and Johannes Fürnkranz. 2008. Efficient pairwise multilabel classification for large-scale problems in the legal domain. In Walter Daelemans, Bart Goethals, and Katharina Morik, editors, *ECML/PKDD (2)*, volume 5212 of *Lecture Notes in Computer Science*, pages 50–65. Springer.

Rada Mihalcea and Andras Csomai. 2007. Wikify!: linking documents to encyclopedic knowledge. In *CIKM '07*, New York, NY, USA. ACM.

Zsolt Minier, Zalan Bodo, and Lehel Csato. 2007. Wikipedia-based kernels for text categorization. In *SYNASC '07*, Washington, DC, USA. IEEE Computer Society.

J. Platt. 2000. Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. In *Advances in Large Margin Classifiers*.

S. Shalev-Shwartz, Y. Singer, and N. Srebro. 2007. Pegasos: Primal estimated sub–gradient solver for SVM. In *Proceedings of the International Conference on Machine Learning*.

G. Tsoumakas, I. Katakis, and I. Vlahavas. 2009. Mining multi-label data. unpublished book chapter.

Pu Wang and Carlotta Domeniconi. 2008. Building semantic kernels for text classification using Wikipedia. In *KDD '08*, New York, NY, USA. ACM.

Yiming Yang. 2001. A study on thresholding strategies for text categorization. In W. Bruce Croft, David J. Harper, Donald H. Kraft, and Justin Zobel, editors, *SIGIR 2001*, pages 137–145. ACM.