The Power of Naïve Query Segmentation

Matthias Hagen Martin Potthast Benno Stein Christof Bräutigam

Bauhaus-Universität Weimar
99421 Weimar, Germany
<first name>.<last name>@uni-weimar.de

ABSTRACT

We address the problem of query segmentation: given a keyword query submitted to a search engine, the task is to group the keywords into phrases, if possible. Previous approaches to the problem achieve good segmentation performance on a gold standard but are fairly intricate. Our method is easy to implement and comes with a comparable accuracy.

Categories and Subject Descriptors: H.3.3 [Information Search and Retrieval]: Query formulation

General Terms: Algorithms, Experimentation

Keywords: Query Segmentation

1. INTRODUCTION

Most Web search queries are sequences of keywords that may comprise complete phrases or compound concepts, e.g., `new york yankees`. Such phrases and concepts can be exploited by search engines as indivisible units in order to improve retrieval precision, or to allow for query reformulation on the level of phrases instead of keywords. Ideally, Web searchers would assist the engines by enclosing their phrases in quotes, but experience shows that many searchers aren't even aware of this option. Hence, search engines apply pre-retrieval algorithms that automatically segment queries in order to second-guess the user's intended phrases and to improve the overall user experience. Our contribution in this respect is a new and simple approach to the task of query segmentation that achieves a segmentation accuracy comparable to state-of-the-art algorithms.

Related Work. Recent research suggests a variety of approaches to query segmentation. For instance, Bendersky et al. [1] use a two-stage procedure. Guo et al. [4] and Yu and Shi [9] use methods based on conditional random fields. However, Yu and Shi explicitly focus on query segmentation in the context of text stored in relational databases and use database-specific features that are not applicable in the Web setting. One of the earliest approaches to Web query segmentation is by Risvik et al. [6]. They segment queries by computing so-called connexity scores that measure mutual information within a segment and the segment frequency in a query log. Jones et al. [5] also use a mutual-information-based scoring that finds segments in which adjacent terms have high mutual information. However, neither Risvik et al. nor Jones et al. experimentally evaluate the segmentation accuracy of their approaches. In more recent papers, methods based on mutual information are used as baselines and they are shown not to perform as good as more involved methods, such as the supervised learning method by Bergsma and Wang [2]. Bergsma and Wang use statistical features obtained by Web queries and from query logs, as well as dependency features that are focused on noun phrases. They also established a gold standard corpus of 500 queries, each segmented by three human annotators. Subsequent work [8, 10] has adopted the corpus, as do we in our evaluations. Bergsma and Wang's supervised learning method is trained on queries segmented by the same annotator who also segmented the gold standard queries. This leaves some doubts with regard to their otherwise remarkably good accuracy. Instead of the supervised approach that requires training data, Tan and Peng [8] and Zhang et al. [10] suggest unsupervised methods based on expectation maximization and eigenspace similarity. Tan and Peng's method, like ours, uses n-gram frequency counts from a large Web corpus, but unlike our method, it tries to establish segment scores via expectation maximization. In a second step they also check whether segments are prominently used in Wikipedia. Instead, Zhang et al. suggest to compute segment scores from the eigenvalues of a correlation matrix corresponding to the query. Hence, all three approaches, Bergsma and Wang's, Tan and Peng's, and Zhang et al.'s, rely on intricate models and techniques whose optimization involves several hyperparameters. By contrast, our approach to query segmentation implements a straightforward usage of n-gram frequency counts, which performs just as well.

2. N-GRAM QUERY SEGMENTATION

The basic and major assumption of our approach is that phrases contained in queries actually exist on the Web. The idea then is to use the Web itself as a corpus of potential query phrases. The largest obtainable collection of Web phrases is the Google n-gram corpus [3]; it contains n-grams of length 1 to 5 from the 2006 Google index along with occurrence frequencies. Based on these Web occurrence frequencies our approach scores a query's possible segmentations and outputs the "best" choice.

We regard a query q as a sequence $(w_1, w_2, \ldots, w_n)$ of n keywords. A valid segmentation S for q is a sequence of disjunct segments s, each a contiguous subsequence of q, whose concatenation equals q. There are 2^{n-1} valid segmentations for q, and $(n^2-n)/2$ potential segments that contain at least two keywords from q. Our algorithm derives a score for a valid segmentation as follows. First, the n-gram frequency $count(s)$ of each potential segment s is retrieved. For n-grams up to $n = 5$ the frequencies can be obtained directly from the corpus; for longer n-grams up to $n = 9$ estimations are made analogously to the set-based method described in [8]. Having the frequencies at hand, all valid segmentations are enumerated systematically, and for each segmentation S a score is

SIGIR'10, July 19–23, 2010, Geneva, Switzerland.
ACM 978-1-60558-896-4/10/07.

Table 1: Segmentation performance on the gold standard.

Annotator	Accuracy Measure	Algorithm MI	[2]	[8]	[10]	Naïve
A	query	0.274	0.638	0.526		0.536
	break	0.693	0.863	0.810		0.807
	seg prec	0.469		0.657	0.652	0.665
	seg rec	0.534		0.657	0.699	0.708
	seg F	0.499		0.657	0.675	0.686
B	query	0.244		0.494		0.380
	break	0.634		0.802		0.752
	seg prec	0.408		0.623	0.632	0.519
	seg rec	0.472		0.640	0.659	0.626
	seg F	0.438		0.631	0.645	0.568
C	query	0.264		0.494		0.454
	break	0.666		0.796		0.772
	seg prec	0.451		0.634	0.614	0.581
	seg rec	0.519		0.642	0.649	0.653
	seg F	0.483		0.638	0.631	0.615
Agree	query	0.343	0.717	0.671		0.627
	break	0.728	0.892	0.871		0.851
	seg prec	0.510		0.767	0.772	0.718
	seg rec	0.550		0.782	0.826	0.778
	seg F	0.530		0.774	0.798	0.746

computed according to the following function:

$$score(S) = \sum_{s \in S, |s| \geq 2} |s|^{|s|} \cdot count(s).$$

The factor $|s|^{|s|}$ gives significant weight to long segments compared to shorter ones in order to compensate the power law distribution of occurrence frequencies on the Web. For example, `"new york"` has a much larger count than `"new york yankees"`, so that the exponential scoring function helps us to avoid segmentations like `"new york" "yankees"`. For a query q we choose from all valid segmentations the segmentation S that maximizes $score(S)$. This naïve approach is competitive with the more involved methods, as our evaluation shows.

3. EVALUATION

We have indexed the Google n-gram corpus in an inverted file in a way similar to [7]. As a query corpus we use the gold standard for query segmentation established by Bergsma and Wang [2], which was also used in subsequent evaluations [8, 10]. Table 1 contains the results reported on that corpus for the mutual information (MI) baseline of [8], the results of the best performing methods from [2, 8, 10], and the results of our naïve approach. The table should be read as follows. Three annotators—A, B, and C—independently segmented the 500 queries of the gold standard, which were originally drawn from the AOL 2006 query log. The annotators segmented 220 of the 500 queries in the same way, denoted in the row named "Agree." As for the segmentation accuracy measures, we report performances on the following three levels: the *query accuracy* is the ratio of segmented queries that match the gold standard, the *break accuracy* is the ratio of decisions between two consecutive words (different/same segment) that match the gold standard, and, at the *segment level* we measure how well the segments found match the gold standard by means of segment precision, segment recall, and segment F-Measure.

The results in Table 1 show that the approach of Bergsma and Wang [2] is slightly better than the other approaches on annotator A as well as on the queries all annotators agree upon. However, it should be noted that their approach is based on a supervised learning algorithm that was explicitly trained on queries segmented by annotator A (the agreement queries also match A's segmentation), and that it requires a lot of additional Web search queries in order to segment one query. Bergsma and Wang did not report exact performance values for the annotators B and C and they did not measure segment level performance. Zhang et al. [10] did not report performances on the query or the break level. As for the two approaches of Tan and Peng [8] and Zhang et al. [10], note that our method is marginally better on annotator A, approximately 0.1 worse on annotator B, and in a 0.05 range on annotator C as well as the agreement queries. Our performance on annotator B can be improved in a postprocessing step which looks for single keyword segments in the segmentation with the highest score, and which adds them to the respective left or right segment if the occurrence frequency $count(s)$ of the resulting segment s is above a threshold of 1000. This tweak then also achieves a 0.05 range on annotator B (query: 0.442; break: 0.774; seg F: 0.579). However, we do not wish to add this tweak to our method for reasons of simplicity, and more importantly, because it appears to be overfitted to annotator B.

As for runtime performance, we are able to segment more than 3000 queries per second on a single machine with sufficient main memory to store all the n-gram counts (about 12 GB).

4. CONCLUSION AND OUTLOOK

In this paper, we presented an approach to query segmentation that is competitive with the best algorithms developed so far, while being less intricate at the same time. As for future work, the evaluation of the state-of-the-art-approaches with respect to gains in retrieval performance is an interesting open problem. To measure segmentation accuracy on a larger scale we are also working on a gold standard that includes significantly more queries.

5. REFERENCES

[1] M. Bendersky, W. B. Croft, and D. Smith. Two-stage query segmentation for information retrieval. In *Proc. of SIGIR 2009*, pages 810–811.

[2] S. Bergsma and Q. I. Wang. Learning noun phrase query segmentation. In *Proc. of EMNLP-CoNLL 2007*, pages 819–826.

[3] T. Brants and A. Franz. Web 1T 5-gram Version 1. Linguistic Data Consortium LDC2006T13, 2006.

[4] J. Guo, G. Xu, H. Li, and X. Cheng. A unified and discriminative model for query refinement. In *Proc. of SIGIR 2008*, pages 379–386.

[5] R. Jones, B. Rey, O. Madani, and W. Greiner. Generating query substitutions. In *Proc. of WWW 2006*, pages 387–396.

[6] K. M. Risvik, T. Mikolajewski, and P. Boros. Query segmentation for Web search. In *Proc. of WWW 2003 (Posters)*.

[7] B. Stein, M. Potthast, and M. Trenkmann. Retrieving customary Web language to assist writers. In *Proc. of ECIR 2010*, pages 631–635.

[8] B. Tan and F. Peng. Unsupervised query segmentation using generative language models and Wikipedia. In *Proc. of WWW 2008*, pages 347–356.

[9] X. Yu and H. Shi. Query segmentation using conditional random fields. In *Proc. of KEYS 2009*, pages 21–26.

[10] C. Zhang, N. Sun, X. Hu, T. Huang, and T.-S. Chua. Query segmentation based on eigenspace similarity. In *Proc. of ACL-IJCNLP 2009*, pages 185–188.