An IR-based Evaluation Framework for Web Search Query Segmentation

Author 1
Author 1 Affiliation
Author 1 Address
Author 1 Email

Author 2
Author 2 Affiliation
Author 2 Address
Author 2 Email

ABSTRACT

This paper presents the first comprehensive evaluation framework for Web search query segmentation based directly on IR performance. In the past, segmentation strategies were mainly validated against manual annotations. Our work shows that the goodness of a segmentation algorithm as judged through evaluation against a handful of human annotated segmentations hardly reflects its effectiveness in an IR-based setup. In fact, state-of the-art algorithms are shown to perform as good as, and sometimes even better than human annotations – a fact masked by previous validations. The proposed framework also provides us an objective understanding of the gap between the present best and the best possible segmentation algorithm. We draw these conclusions based on an extensive evaluation of six segmentation strategies, including three most recent algorithms, vis-à-vis segmentations from three human experts. The evaluation framework also gives insights about which segments should be necessarily detected by an algorithm for achieving the best retrieval results. The meticulously constructed dataset used in our experiments has been made public for use by the research community.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Query formulation, Retrieval models

General Terms

Measurement, Experimentation, Human Factors

Keywords

Query segmentation, IR evaluation, Evaluation framework, Test collections, Manual annotation

1. INTRODUCTION

Query segmentation is the process of dividing a query into individual semantic units [3]. For example, the query `singular value decomposition online demo` can be broken into `singular value decomposition` and `online demo`. All documents containing the individual terms `singular`, `value` and `decomposition` are not necessarily relevant for this query. Rather, one can almost always expect to find the segment `singular value decomposition` in the relevant documents. In contrast, although `online demo` is a segment, finding the phrase or some variant of it may not affect the relevance of the document. Hence, the potential of query segmentation goes beyond the detection of multiword named entities. Rather, segmentation leads to a better understanding of the query and is crucial to the search engine for improving Information Retrieval (IR) performance.

There is broad consensus in the literature that query segmentation can lead to better retrieval performance [2, 3, 10, 7, 15]. However, most automatic segmentation techniques [3, 4, 7, 10, 15, 18] have so far been evaluated only against a small set of 500 queries segmented by human annotators. Such an approach implicitly assumes that a segmentation technique that scores better against human annotations will also automatically lead to better IR performance. We challenge this approach on multiple counts. First, there has been no systematic study that establishes the quality of human segmentations in the context of IR performance. Second, grammatical structure in queries is not as well-understood as natural language sentences where human annotations have proved useful for training and testing of various Natural Language Processing (NLP) tools. This leads to considerable inter-annotator disagreement when humans segment search queries. Third, good quality human annotations for segmentation can be difficult and expensive to obtain for a large set of test queries. Thus, there is a need for a more direct IR-based evaluation framework for assessing query segmentation algorithms. This is the central motivation of the present work.

We propose an IR-based evaluation framework for query segmentation that requires only human relevance judgments for query-URL pairs for computing the performance of a segmentation algorithm – such relevance judgments are often available in plenty because they are anyway needed for training and testing of any IR engine. A fundamental problem in designing an IR-based evaluation framework for segmentation algorithms is to decouple the effect of segmentation accuracy from the way segmentation is used for IR. This is because a query segmentation algorithm breaks the input query into, typically, a non-overlapping sequence of words (segments), but it does not prescribe how these segments should be used during the retrieval and ranking of the docu-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
SIGIR '12 Portland, Oregon
Copyright 20XX ACM X-XXXXX-XX-X/XX/XX ...$10.00.

ments for that query. We resolve this problem by providing a formal model of query expansion for a given segmentation; the various queries obtained can then be issued to any standard IR engine, which we assume to be a black-box.

We conduct extensive experiments within our framework to understand the performance of several state-of-the-art query segmentation schemes [7, 10, 12] and segmentations by three human experts. Our experiments reveal several interesting facts such as: (a) Segmentation is actively useful in improving IR performance, even though submitting all segments (detected by an algorithm) in double quotes to the IR engine degrades performance; (b) All segmentation strategies, including human segmentations, are yet to reach the best achievable limits in IR performance; (c) Human segmentations do not always coincide with the ideal segmentations that maximize IR performance; (d) In terms of IR metrics, some of the segmentation algorithms perform as good as the best human annotator and better than the average/worst human annotator; (e) Current match-based metrics for comparing query segmentation against human annotations are only weakly correlated with the IR-based metrics, and cannot be used as a proxy for IR performance; and (f) There is a need for defining an appropriate metric to compare segmentations against human annotations that differentially penalizes splitting and joining of reference segments. In short, the proposed evaluation framework not only provides a formal way to compare segmentation algorithms and estimate their effectiveness in IR, but also helps us to understand the gaps in human annotation-based evaluation. The framework also provides valuable insights regarding the segmentations that can be used for improvement of the algorithms.

The rest of the paper is organized as follows. Sec. 2 introduces our evaluation framework and its design philosophy. Sec. 3 presents the dataset and the segmentation algorithms compared on our framework. Sec. 4 discusses the experimental results and insights derived from them. In Sec. 5, we discuss a few related issues, and the next section (Sec. 6) gives a brief background of past approaches to evaluate query segmentation and their limitations. We conclude by summarizing our contributions and suggesting future work in Sec. 7.

2. THE EVALUATION FRAMEWORK

In this section we present a framework for the evaluation of query segmentation algorithms based on IR performance. Let $\mathbf{q}$ denote a search query and let $\mathbf{s}^{\mathbf{q}} = \langle s_1^{\mathbf{q}}, \ldots, s_n^{\mathbf{q}} \rangle$ denote a segmentation of $\mathbf{q}$ such that a simple concatenation of the segments equals $\mathbf{q}$, i.e., we have $\mathbf{q} = (s_1^{\mathbf{q}} + \cdots + s_n^{\mathbf{q}})$, where $+$ represents the concatenation operator. We are given a segmentation algorithm $\mathcal{A}$ and the task is to evaluate its performance. We require the following resources:

1. A test set $\mathcal{Q}$ of unquoted search queries.
2. A set $\mathcal{U}$ of documents (or URLs) out of which search results will be retrieved.
3. Relevance judgments $r(\mathbf{q}, u)$ for query-URL pairs $(\mathbf{q}, u) \in \mathcal{Q} \times \mathcal{U}$. The set of all relevance judgments are collectively denoted by $\mathcal{R}$.
4. An IR engine that supports quoted queries as input.

Table 1: Example of generation of quoted versions for a segmented query.

Segmented query	Quoted versions
`we are | the people | song lyrics`	`we are the people song lyrics`
	`we are the people "song lyrics"`
	`we are "the people" song lyrics`
	`we are "the people" "song lyrics"`
	`"we are" the people song lyrics`
	`"we are" the people "song lyrics"`
	`"we are" "the people" song lyrics`
	`"we are" "the people" "song lyrics"`

The resources needed by our evaluation framework are essentially the same as those needed for the training and testing of a standard IR engine, namely, queries, a document corpus and set of relevance judgments. Akin to the training examples required for an IR engine, we only require relevance judgments for a small (appropriate) subset of $\mathcal{Q} \times \mathcal{U}$ (each query needs only the documents in its own pool to be judged) [16].

It is useful to separate the evaluation of segmentation performance, from the question of how to best exploit the segments to retrieve the most relevant documents. From an IR perspective, a natural interpretation of a segment could be that it consists of words that must appear together, in the same order, in documents where the segment is deemed to match (see, for instance, [3]). This can be referred to as *ordered contiguity matching*. While this can be easily enforced in modern IR engines through use of double quotes around segments, we observe that not all segments must be used this way (see [11] for similar ideas and related experiments even though in a different context). Some segments may admit more general matching criteria, such as *unordered or intruded contiguity* (e.g., a segment `a b` may be allowed to match `b a` or `a c b` in the document). The case of unordered intruded matching may be restricted under *linguistic dependence* assumptions (e.g., `a b` can match `a` *`of`* `b` or `b` *`in`* `a`). Finally, some segments may even play non-matching roles (e.g., when the segment specifies preferred user intent, like `how to` and `where is`). Thus, there may be several different ways to exploit the segments discovered by a segmentation algorithm. Even within the same query, different segments may require to be treated differently. For instance, in the query `cannot view | word files | windows 7`, the first one might be matched using intruded ordered occurrence (`cannot` *`properly`* `view`), the second segment may be matched under a linguistic dependency model (`files` *`in`* `word`) and the last one under ordered contiguity.

Intruded contiguity and linguistic dependency may be difficult to implement for the broad class of general Web search queries. Identifying how the various segments of a query should be ideally matched in the document is quite a challenging and unsolved research problem. On the other hand, an exhaustive expansion scheme, where every segment is expanded in every possible way, is computationally expensive and might introduce noise. Moreover, current commercial IR engines do not support any syntax to specify linguistic dependence or intruded or unordered occurrence based matching. Hence, in order to keep the evaluation framework in line with the current commercial IR systems, we focus on ordered contiguity matching which is easily implemented through the use of double quotes around segments in modern IR engines. However, we note that the philosophy of the framework does not change with increased sophistication in the retrieval system – only the expansion sets for the queries have to be appropriately modified.

We propose an evaluation framework for segmentation algorithms that generates all possible quoted versions of a segmented query (see Table 1) and submits each quoted version to the IR engine. The corresponding ranked lists of retrieved documents are then assessed against relevance judgments available for the query-URL pairs. The IR quality of the best-performing quoted version is used to measure performance of the segmentation algorithm. We now formally specify our evaluation framework that computes what we call a Quoted Version Retrieval Score (QVRS) for the segmentation algorithm given the test set $\mathcal{Q}$ of queries, the document pool $\mathcal{U}$ and the relevance judgments $\mathcal{R}$ for query-URL pairs.

Quoted query version generation

Let the segmentation output by algorithm $\mathcal{A}$ be denoted by $\mathcal{A}(\mathbf{q}) = \mathbf{s}^{\mathbf{q}} = \langle s_0^{\mathbf{q}}, \ldots, s_{n-1}^{\mathbf{q}} \rangle$. We generate all possible *quoted versions* of the query $\mathbf{q}$ based on the segments in $\mathcal{A}(\mathbf{q})$. In particular, we define $\mathcal{A}_0(\mathbf{q}) = (s_1^{\mathbf{q}} + \cdots + s_n^{\mathbf{q}})$ with no quotes on any of the segments, $\mathcal{A}_1(\mathbf{q}) = (s_1^{\mathbf{q}} + \cdots + {''}s_n^{\mathbf{q}}{''})$ with quotes only around the last segment $s_n^{\mathbf{q}}$, and so on. Since there are n segments in $\mathcal{A}(\mathbf{q})$, this process will generate 2^n versions of the query, $\mathcal{A}_i(\mathbf{q})$, $i = 0, \ldots, 2^n - 1$. We note that if $b_i = (b_{i1}, \ldots, b_{in})$ is the n-bit binary representation of i, then $\mathcal{A}_i(\mathbf{q})$ will apply quotes to the j^{th} segment $s_j^{\mathbf{q}}$ if and only if $b_{ij} = 1$. We deduplicate this set, because $\{\mathcal{A}_i(\mathbf{q}) : i = 0, \ldots, 2^n - 1\}$ can contain multiple versions that essentially represent the same quoted query version (for example, if single words are inside quotes) in terms of the input semantics of an IR engine. The resulting set of unique quoted query versions is denoted $\mathcal{Q}_{\mathcal{A}}(\mathbf{q})$.

Document retrieval using IR engine

For each $\mathcal{A}_i(\mathbf{q}) \in \mathcal{Q}_{\mathcal{A}}(\mathbf{q})$ we use the IR engine to retrieve a ranked list $\mathcal{O}_i$ of documents out of the document pool $\mathcal{U}$ that matched the given quoted query version $\mathcal{A}_i(\mathbf{q})$. The number of documents retrieved in each case depends on the IR metrics we will want to use to assess the quality of retrieval. For example, to compute an IR metric at the top k positions, we would require that at least k documents be retrieved from the pool. Our evaluation framework is relevant given any retrieval engine that can support quoted queries as input.

Measuring retrieval against relevance judgments

Since we have relevance judgments ($\mathcal{R}$) for query-URL pairs in $\mathcal{Q} \times \mathcal{U}$, we can now compute IR metrics such as normalized Discounted Cumulative Gain (nDCG) [8], Mean Average Precision (MAP) [14] or Mean Reciprocal Rank (MRR) [17] to measure the quality of the retrieved ranked list $\mathcal{O}_i$ for query $\mathbf{q}$. We use @k variants of each of these measures which are defined to be the usual metrics computed after examining only the top k positions. For example, we can compute nDCG@k for query $\mathbf{q}$ and retrieved document-list $\mathcal{O}_i$ using the following formula:

$$\text{nDCG@}k(\mathbf{q}, \mathcal{O}_i\,;\,\mathcal{R}) = r(\mathbf{q}, \mathcal{O}_i^1) + \sum_{j=2}^{k} \frac{r(\mathbf{q}, \mathcal{O}_i^j)}{\log_2 j} \qquad (1)$$

where $\mathcal{O}_i^j$, $j = 1, \ldots, k$, denotes the j^{th} document in the ranked-list $\mathcal{O}_i$ and $r(\mathbf{q}, \mathcal{O}_i^j)$ denotes the associated relevance judgment from $\mathcal{R}$.

Oracle score using best quoted query version

Different quoted query versions $\mathcal{A}_i(\mathbf{q})$ (all derived from the same basic segmentation $\mathcal{A}(\mathbf{q})$ output by the segmentation algorithm $\mathcal{A}$) retrieve different ranked lists of documents $\mathcal{O}_i$. As discussed earlier, automatic apriori selection of a good (or the best) quoted query version is a difficult problem. While different strategies may be used to select a quoted query version, we would like our evaluation of the segmentation algorithm $\mathcal{A}$ to be agnostic of the version-selection step. To this end, we select the best-performing $\mathcal{A}_i(\mathbf{q})$ from the entire set $\mathcal{Q}_{\mathcal{A}}(\mathbf{q})$ of query versions generated and use it to define our *oracle score* for $\mathbf{q}$ and $\mathcal{A}$ under the chosen IR metric. For example, the oracle score for nDCG@k is as defined below:

$$\Omega_{\text{nDCG@}k}(\mathbf{q}, \mathcal{A}) = \max_{\mathcal{A}_i(\mathbf{q}) \in \mathcal{Q}_{\mathcal{A}}(\mathbf{q})} \text{nDCG@}k(\mathbf{q}, \mathcal{O}_i\,;\,\mathcal{R}) \qquad (2)$$

where $\mathcal{O}_i$ denotes the ranked list of documents retrieved by the IR engine when presented with $\mathcal{A}_i(\mathbf{q})$ as the input. We note that $\mathcal{Q}_{\mathcal{A}}(\mathbf{q})$ always contains the original unsegmented version of the query. We refer to such an $\Omega_{\cdot}(\cdot, \cdot)$ as the *Oracle*.

This forms the basis of our evaluation framework. We note that there can also be other ways to define this oracle score. For example, instead of seeking the best IR performance possible across the different query versions, we could also seek the minimum performance achievable by $\mathcal{A}$ irrespective of what version-selection strategy is adopted. This would give us a lower bound on the performance of the segmentation algorithm. However, the main drawback of this approach is that the minimum performance is almost always achieved by the fully quoted version (where every segment is in double quotes). Such a lower bound would not be useful in assessing the comparative performance of segmentation algorithms.

QVRS computation

Once the oracle scores are obtained for all queries in the test set $\mathcal{Q}$, we can compute the average oracle score achieved by $\mathcal{A}$. We refer to this as the Quoted Version Retrieval Score (QVRS) of $\mathcal{A}$ with respect to test set $\mathcal{Q}$, document pool $\mathcal{U}$ and relevance judgments $\mathcal{R}$. For example, using the oracle with the nDCG@k metric, we can define the QVRS score as follows:

$$QVRS(\mathcal{Q}, \mathcal{A}, \text{nDCG@}k) = \frac{1}{|\mathcal{Q}|} \sum_{\mathbf{q} \in \mathcal{Q}} \Omega_{\text{nDCG@}k}(\mathbf{q}, \mathcal{A}) \qquad (3)$$

Similar QVRS scores can be computed using other IR metrics such as MAP@k and MRR@k. In our experiments section, we report results using nDCG@k, MAP@k, and MRR@k, for $k = 5$ and $k = 10$ as most Web users examine only the first five or ten search results.

3. DATASET AND ALGORITHMS

In this section, we describe the dataset used and briefly introduce the algorithms included in our evaluation. First, we provide details of the test set $\mathcal{Q}$ of queries in Sec. 3.1. In Sec. 3.2 we describe the methodology used for collecting the pool of documents $\mathcal{U}$. In Sec. 3.3, we provide brief descriptions of the segmentation algorithms used in our evaluation.

3.1 Test set of queries ($\mathcal{Q}$)

We selected a random subset of 500 queries from query logs of a commercial Web search engine containing 16.7 million queries issued over a period of one month (May – June 2010). We used the following criteria to filter the logs before extracting a random sample: (1) Exclude queries with non-ASCII characters, (2) Exclude queries that occurred less than 5 times in the logs (rarer queries often contained spelling errors), and (3) Restrict query lengths to between five and eight words. Shorter queries rarely contain multiple multiword segments, and when they do, they are mostly named entities that can be easily detected using dictionaries. Moreover, traditional search engines usually give satisfactory results for short queries. On the other hand, queries longer than eight words (only 3.24% of all queries in our logs) are usually error messages, complete NL sentences or song lyrics.

We denote this set of 500 queries by $\mathcal{Q}$, the test set of unquoted queries needed for all our evaluation experiments. The average length of queries in $\mathcal{Q}$ is 5.29 words. This was 4.31 words in the Bergsma and Wang 2007 [3] Corpus[1] (henceforth, BWC07). Each of these 500 queries were independently segmented by three human annotators (who issue around 20-30 search queries per day and rate their own search expertise as high) who were asked to mark a contiguous chunk of words in a query as a *segment* if they thought that these words together formed a coherent semantic unit. The annotators were free to refer to other resources and Web search engines during the annotation process, especially for understanding the query and its possible context(s). We shall refer to the three sets of annotations (and also the corresponding annotators) as H_A, H_B and H_C.

It is important to mention that the queries in $\mathcal{Q}$ have some amount of word level overlap, even though all the queries have very distinct information needs. Thus, a document retrieved from the pool might exhibit good term level match for more than one query in $\mathcal{Q}$. This makes our corpus an interesting testbed for experimenting with different retrieval systems as well. There are existing datasets, including BWC07, that could have been used for this study. However, refer to Sec. 5.1 for an account of why building this new dataset was crucial for our research.

3.2 Document pool ($\mathcal{U}$)

Each query in $\mathcal{Q}$ was segmented using all the segmentation algorithms considered in our study (see Sec. 3.3). For every segmentation, all possible quoted versions were generated and then submitted to the Bing API[2] (Google has discontinued its Web search API service since November 2010) and the top ten documents were retrieved. We then deduplicated these URLs to obtain $14,171$ unique URLs, forming $\mathcal{U}$. We observed that for 71.4% of the queries there is less than 50% overlap between the top-10 URLs retrieved for the different quoted versions. This indicates that different ways of quoting the segments in a query does make a difference in the search results. By varying the pooling depth (ten in our case), one can roughly control the number of relevant and non-relevant documents entering the collection.

For each query-URL pair, where the URL has been retrieved for at least one of the quoted versions of the query (approx. 28 per query), we obtained three independent sets of relevance judgments from human experts. These experts were different from annotators H_A, H_B and H_C who marked the segmentations. For each query, the corresponding set of URLs was shown to experts after deduplication and randomization (to prevent position bias for top results), and asked to mark whether the URL was *irrelevant* (score = 0), *partially relevant* (score = 1) or *highly relevant* (score = 2) to the query. We then computed the average rating for each query-URL pair (the entire set forming $\mathcal{R}$), which has been used for subsequent nDCG, MAP and MRR computations. Please refer to Table 8 in Sec. 5.3 for inter-annotator agreement figures and other related discussions.

Table 2: Segmentation algorithms compared on our framework.

Algorithm	Training data
Li et al. [10]	Click data, Web n-gram probabilities
Hagen et al. [7]	Web n-gram frequencies, Wikipedia titles
Mishra et al. [12]	Query logs
[12] + Wiki	Query logs, Wikipedia titles
PMI-W [7]	Web n-gram probabilities (used as baseline)
PMI-Q [12]	Query logs (used as baseline)

3.3 Segmentation algorithms

Table 2 lists the six segmentation algorithms that have been studied in this work. Li et al. [10] use the expectation maximization algorithm to arrive at the most probable segmentation, while Hagen et al. [7] show a simple frequency-based method produces a performance comparable to the state-of-the-art. The technique in [12] uses only query logs for segmenting queries. In our experiments, we observed that the performance of [12] can be improved if we used Wikipedia titles. We refer to this as "[12] + Wiki" in our experiments (see Appendix A for details). The Point-wise Mutual Information (PMI)-based algorithms are used as baselines. The thresholds for PMI-W and PMI-Q were chosen to be 8.141 and 0.156 respectively, that maximized the $Seg-F$ (see Sec. 4.2) on our development set.

3.4 Public release of data

The test set of search queries along with their manual and some of the algorithmic segmentations, the theoretical best segmentation output that can serve as an evaluation benchmark ((BQV_{BF}) in Sec. 4.1), and the list of URLs whose contents serve as our document corpus is available for public use[3]. The relevance judgments for the query-URL pairs have also been made public which will enable the community to use this dataset for evaluation of any new segmentation algorithm.

4. EXPERIMENTS AND OBSERVATIONS

In this section we present experiments, results and the key inferences made from them.

4.1 IR Experiments

For the retrieval-based evaluation experiments, we use the Lucene[4] text retrieval system, which is publicly available as a code library. In its default configuration, Lucene does

[1] `http://bit.ly/xoyT2c`

[2] `http://msdn.microsoft.com/en-us/library/dd251056.aspx`

[3] `https://querysegmentation.wikispaces.com/`

[4] `http://lucene.apache.org/java/docs/index.html`

Table 3: Results of IR-based evaluation of segmentation algorithms using Lucene.

Metric	Unseg. query	[10]	[7]	[12]	[12] + Wiki	PMI-W	PMI-Q	H_A	H_B	H_C	BQV_{BF}
nDCG@5	0.688	0.752*	**0.763***	0.745	**0.771***	0.691	**0.766***	**0.770**	0.768	0.759	0.802*
nDCG@10	0.701	0.756*	**0.767***	0.751	**0.771***	0.704	**0.767***	**0.770**	**0.768**	**0.763**	0.813*
MAP@5	0.882	0.930*	**0.942***	0.930*	**0.946***	0.884	0.932*	**0.944**	**0.942**	0.936	0.950*
MAP@10	0.865	0.910*	**0.921***	0.910*	**0.924***	0.867	0.912*	**0.923**	**0.921**	**0.916**	0.935*
MRR@5	0.538	0.632*	**0.649***	0.609	**0.657***	0.543	**0.648***	**0.656**	**0.648**	0.632	0.716*
MRR@10	0.549	0.640*	**0.658***	0.619	**0.665***	0.555	**0.656***	**0.665**	**0.656**	0.640	0.724*

The highest value in a row (excluding the BQV_{BF} column) and those with no statistically significant difference with the highest value are marked in **boldface**. The values for algorithms that perform better than or have no statistically significant difference with the *minimum* of the human segmentations are marked with *. The paired t-test was performed and the null hypothesis was rejected if the p-value was less than 0.05.

Table 4: Matching metrics for different segmentation algorithms and human annotations *with* BQV_{BF} *as reference.*

Metric	Unseg. query	[10]	[7]	[12]	[12] + Wiki	PMI-W	PMI-Q	H_A	H_B	H_C	BQV_{BF}
Qry-Acc	0.036	0.076*	0.064*	0.056*	0.070*	0.034	**0.112***	0.062	0.068	0.064	1.000
Seg-Prec	**0.248***	0.189*	0.178*	0.220*	0.162*	**0.247***	0.210*	0.166	0.164	0.179	1.000
Seg-Rec	**0.366***	0.181*	0.152*	0.232*	0.128*	**0.359***	0.191*	0.130	0.130	0.159	1.000
Seg-F	**0.296***	0.183*	0.164	0.226	0.143	**0.293**	0.200	0.146	0.145	0.168	1.000
Seg-Acc	0.490	0.377	0.646*	0.606	**0.657***	0.494	0.642*	**0.656**	**0.658**	0.647	1.000

The highest value in a row (excluding the BQV_{BF} column) and those with no statistically significant difference with the highest value are marked in **boldface**. The values for algorithms that perform better than or have no statistically significant difference with the *minimum* of the human segmentations are marked with *. The paired t-test was performed and the null hypothesis was rejected if the p-value was less than 0.05.

not perform any automatic query segmentation, which is very important for examining the effectiveness of segmentation algorithms in an IR-based scheme. Double quotes can be used in a query to force Lucene to match the quoted *phrase* (in Lucene terms) exactly in the documents. Starting with the segmentations output by each of the six algorithms as well as the three human annotations, we generated all possible quoted query versions, which resulted in a total of $5,562$ versions for the 500 queries. In the notation of Sec. 2, this corresponds to generating $\mathcal{Q}_{\mathcal{A}}(\mathbf{q})$ for each segmentation method $\mathcal{A}$ (including one for each human segmentation) and for every query $\mathbf{q} \in \mathcal{Q}$. These quoted versions were then passed through Lucene to retrieve documents from the pool. For each segmentation scheme, we then use the oracle described in Sec. 2 to obtain the query version yielding the best result (as determined by the IR metrics – nDCG, MAP and MRR computed according to the human relevance judgments). These oracle scores are then averaged over the query set to give us the QVRS measures.

The results are summarized in Table 3. Different rows represent the different IR metrics that were used and columns correspond to different segmentation strategies. Specifically, the second column (marked "Unseg. Query") refers to the original unsegmented query, columns 3-8 denote the six different segmentation algorithms, columns 9-11 (marked H_A, H_A and H_A) represent the human segmentations, and the last column represents the performance of the best quoted versions (denoted by BQV_{BF} in table) of the queries which are computed by a *brute force* or exhaustive search over all possible ways of quoting the parts of a query (2^{k-1} possible quoted versions for a k-word query) irrespective of any segmentation algorithm. The results are reported for two sizes of retrieved URL lists (k), namely five and ten. Since we needed to convert our graded relevance judgments to binary values for computing MAP@k, URLs with ratings of 1 and 2 were considered as relevant (responsible for the generally high values) and those with 0 as irrelevant. For MRR, only URLs with ratings of 2 were considered as "relevant".

The first observation we make from the results is that human as well as all algorithmic segmentation schemes consistently outperform unsegmented queries for all IR metrics. Second, we observe that the performance of some segmentation algorithms are comparable and sometime even marginally better than human annotations. Finally, we observe that there is considerable scope for improving IR performance through better segmentation (all values less than BQV_{BF}). The inferences from these observations are stated later in this section.

4.2 Performance under traditional matching metrics

In the next set of experiments we study the utility of traditional matching metrics that are used to evaluate query segmentation algorithms against a gold standard of human segmentated queries (also known as *reference* segmentation). These metrics are listed below (For a more detailed discussion on matching metrics, see Sec. 6.1 of [7]):

1. **Query accuracy (*Qry-Acc*):** The fraction of queries where the output matches exactly with the reference segmentation.

2. **Segment precision (*Seg-Prec*):** The ratio of the number of segments that overlap in the output and

Table 5: Performance of PMI-Q and [10] with respect to matching (mean of comparisons with H_A, H_B and H_C as references) and IR metrics.

Metric	nDCG@10	MAP@10	MRR@10	Qry-Acc	Seg-Prec	Seg-Rec	Seg-F	Seg-Acc
PMI-Q	**0.767**	**0.912**	**0.656**	0.341	0.448	0.487	0.467	**0.810**
[10]	0.756	0.910	0.640	**0.375**	**0.524**	**0.588**	**0.554**	**0.810**

The highest values in a column are marked in **boldface**.

reference segmentations to the number of output segments, averaged across all queries in the test set.

3. **Segment recall (*Seg-Rec*):** The ratio of the number of segments that overlap in the output and reference segmentations to the number of reference segments, averaged across all queries in the test set.

4. **Segment F-score (*Seg-F*):** The harmonic mean of *Seg-Prec* and *Seg-Rec*.

5. **Segmentation accuracy (*Seg-Acc*):** The ratio of correctly predicted boundaries and non-boundaries in the output segmentation with respect to the reference, averaged across all queries in the test set.

We computed the matching metrics for various segmentation algorithms against H_A, H_B and H_C. According to these metrics, "[12] + Wiki" turns out to be the best algorithm which agrees with the results of IR evaluation. However, the average Kendall-Tau rank correlation coefficient[5] between the ranks of the strategies as obtained from the IR metrics (Table 3) and the matching metrics was only 0.75. This indicates that matching metrics are not perfect predictors for IR performance. In fact, we discovered some costly flaws in the relative ranking produced by matching metrics. One such case was rank inversions between [10] and PMI-Q. The relevant results are shown in Table 5, which demonstrate that while PMI-Q consistently performs better than [10] under IR-based measures, the opposite inference would have been drawn if we had used any of the matching metrics.

In [3], human annotators were asked to segment queries such that segments matched exactly in the relevant documents. This essentially corresponds to determining the best quoted versions for the query. Thus, it would be interesting to study how traditional matching metrics would perform if the humans actually marked the best quoted versions. In order to evaluate this, we used the matching metrics to compare the segmentation outputs by the algorithms and human annotations against BQV_{BF}. The corresponding results are quoted in Table 4. The results show that matching metrics are very poor indicators of IR performance with respect to the BQV_{BF}. For example, for three out of the five matching metrics, the unsegmented query is ranked the best. This shows that even if human annotators managed to correctly guess the best quoted versions, the matching metrics would fail to estimate the correct relative rankings of the segmentation algorithms with respect to IR performance. This fact is also borne out in the Kendall-Tau rank correlation coefficients reported in Table 6. Another interesting observation from these experiments is that *Seg-Acc* emerges as the best matching metric with respect to IR performance, although its correlation coefficient is still much below one.

[5]This coefficient is 1 when there is perfect concordance between the rankings, and -1 if the trends are reversed.

Table 6: Kendall-Tau coefficients between IR and matching metrics *with BQV_{BF} as reference for the latter.*

Metric	Qry-Acc	Seg-Prec	Seg-Rec	Seg-F	Seg-Acc
nDCG@10	0.432	-0.854	-0.886	-0.854	**0.674**
MAP@10	0.322	-0.887	-0.920	-0.887	**0.750**
MRR@10	0.395	-0.782	-0.814	-0.782	**0.598**

The highest value in a row is marked in **boldface**.

4.3 Inferences

Segmentation is helpful for IR. By definition, $\Omega.(\cdot,\cdot)$ (i.e., the oracle) values for every IR metric for any segmentation scheme are at least as large as the corresponding values for the unsegmented query. Nevertheless, for every IR metrics, we observe significant performance benefits for all the human and algorithmic segmentations (except for PMI-W) over the unsegmented query. This indicates that segmentation is indeed helpful for boosting IR performance. Thus, our results validate the prevailing notion and some of the earlier observations [2, 10] that segmentation can help improve IR.

Human segmentations are a good proxy, but not a true gold standard. Our results indicate that human segmentations perform reasonably well in IR metrics. The best of the human annotators beats all the segmentation algorithms, except for "[12] + Wiki", on all the metrics. Therefore, evaluation against human annotations can indeed be considered as the second best alternative to an IR-based evaluation (though see below for criticisms of current matching metrics). However, if the objective is to improve IR performance, then human annotations cannot be considered a true gold standard. There are at least three reasons for this:

First, in terms of IR metrics, some of the state-of-the-art segmentation algorithms are performing as good as human segmentations. Therefore, further optimization of the matching metrics against human annotations is not going to improve (it can only degrade) the IR performance of the segmentation algorithms. Thus, evaluation on human annotations might become a limiting factor for the current segmentation algorithms.

Second, the IR performance of the best quoted version of the queries derived through our framework is significantly better than that of human annotations (last column, Table 3). This means that humans fail to predict the correct boundaries in many instances. Thus, there is scope of improvement for human annotations.

Third, IR performance of at least one of the three human annotators (namely H_C) is poorer than four of the algorithms studied. In other words, while some annotators (such as H_A) are good at guessing the "correct" segment boundaries that will help IR, not all annotators can do it

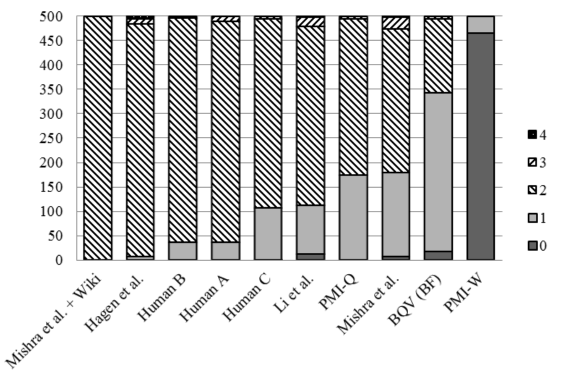

Figure 1: Distribution of multiword segments in queries across segmentation strategies.

well. Therefore, unless the annotators are chosen and guided properly, one cannot guarantee the quality of annotated data for query segmentation. If the queries in the test set have multiple intents, this issue becomes an even bigger concern.

Matching metrics are misleading. As discussed earlier and demonstrated by Tables 4 and 6, the matching metrics provide unreliable ranking of the segmentation algorithms even when applied against a true gold standard, BQV_{BF}, that by definition maximizes IR performance. This counter-intuitive observation can be explained in two ways. Either the matching metrics or the IR metrics (or probably both) are misleading. Given that IR metrics are well-tested and generally assumed to be acceptable, we are forced to conclude that the matching metrics do not really reflect the quality of a segmentation with respect to a gold standard. Indeed, this can be illustrated by a simple example.

Example. Let us consider the query `the looney toons show cartoon network`, whose best quoted version turns out to be `"the looney toons show" "cartoon network"`. The underlying segmentation that can give rise to this and therefore can be assumed to be the reference is:
Ref: `the looney toons show | cartoon network`
The segmentations
(1) `the looney | toons show | cartoon | network`
(2) `the | looney | toons show cartoon | network`
are equally bad if one considers the matching metrics of *Qry-Acc*, *Seg-Prec*, *Seg-Rec* and *Seg-F* (all values being zero) with respect to the reference segmentation. $Seg - Acc$ values for the two segmentations are 3/5 and 1/5 respectively. However, the BQV for (1) (`"the looney" "toons show" cartoon network`) fetches better pages than the BQV of (2) (`the looney toons show cartoon network`). So the segmentation (2) provides no IR benefit over the unsegmented query and hence performs worse than (1) on IR metrics. However, the matching metrics, except for *Seg-Acc* to some extent, fail to capture this difference between the segmentations.

Distribution of multiword segments across queries gives insights about effectiveness of strategy. The limitation of the matching metrics can also be understood from the following analysis of the multiword segments in the queries. Fig. 1 shows the distribution of queries having a specific number of multiword segments (for example, 1 in the legend indicates the proportion of queries having *one* multiword segment) when segmented according to the various schemes. We note that for "Mishra et al. [12] + Wiki", Hagen et al. [7], H_B and H_A, almost all of the queries have two multiword segments. For H_C, Li et al. [10], PMI-Q and Mishra et al. [12], the proportion of queries that have only one multiword segment increases. Finally, PMI-W has almost negligible queries with a multiword segment. BQV_{BF} is different from all of them and has a majority of queries with one multiword segment. Now given that the first group generally does the best in IR, followed by the second, we can say that out of the two multiword segments marked by these strategies, only one needs to be quoted. PMI-W as well as unsegmented queries are bad because these schemes cannot detect the one crucial multiword segment quoting which improves the performance. Nevertheless, these schemes do well for matching metrics against BQV_{BF} because both have a large number of single word segments. Clearly this is not helpful for IR. Finally, Mishra et al. [12] performs poorly despite being able to identify a multiword segment in most of the cases because it is not identifying the one that is important for IR.

Hence, the matching metrics are misleading due to two reasons. First, they do not take into account that splitting a useful segment (i.e., a segment which should be quoted to improve IR performance) is less harmful than joining two unrelated segments. Second, matching metrics are, by definition, agnostic to which segments are useful for IR, and therefore, they might unnecessarily penalize a segmentation for not agreeing on the segments which should not be quoted, but are present in the reference human segmentation. While the latter is an inherent problem with any evaluation against manually segmented datasets, the former can be resolved by introducing a new matching metric that differentially penalizes splitting and joining of segments. This is an important and interesting research problem that we would like to address in the future. However, we would like to emphasize here that with the IR system expected to grow in complexity in the future (supporting more flexible matching criteria), the need for an IR-based evaluation like ours' becomes imperative.

Based on our new evaluation framework and corresponding experiments, we observe that "[12] + Wiki" has the best performance. Nevertheless, the algorithms are trained and tested on different datasets, and therefore, a comparison amongst the algorithms might not be entirely fair. This is not a drawback of the framework and can be circumvented by appropriately training and tuning all the algorithms on similar datasets. However, the objective of the current work is not to compare segmentation algorithms; rather, it is to introduce the evaluation framework, gain insights from the experiments and highlight the drawbacks of human segmentation based evaluation.

5. RELATED ISSUES

In this section, we will briefly discuss a few related issues that are essential for understanding certain design choices and decisions made during the course of this research.

5.1 Motivation for a new dataset

TREC data has been a popular choice for conducting IR-based experiments throughout the past decade. Since there is no track specifically geared towards query segmentation, the queries and *qrels* (query-relevance sets) from the ad hoc

Table 7: IR-based evaluation using Bing API.

Metric	Unseg. query	All quoted for [12] + Wiki	Oracle for [12] + Wiki
nDCG@10	0.882	0.823	**0.989***
MAP@10	0.366	0.352	**0.410***
MRR@10	0.541	0.515	**0.572***

The best value in a row is marked **bold**. Statistically significant ($p < 0.05$ for paired t-test) improvement over the unsegmented query is marked with *.

retrieval task for the Web Track would seem the most relevant to our work. However, 74% of the 50 queries in the 2010 Web track ad hoc task had less than three words. Also, when these 50 queries were segmented using the six algorithms, half of the queries did not have a multiword segment. As discussed earlier, query segmentation is useful but not necessarily for all types of queries. The benefit of segmentation can be observed, if at all, only when there are multiple multiword segments in the queries. The TREC Million Query Track, last held in 2009, has a much larger set of 40,000 queries, with a better coverage of longer queries. But since the goal of the track is to test the hypothesis that a test collection built from several incompletely judged topics is a better tool than a collection built using traditional TREC pooling, there are only about 35,000 query-document relevance judgments for the 40,000 queries. Such a sparse relevance judgment set is not suitable here – incomplete assessments, especially for documents near the top ranks, could cause crucial errors in system comparisons. Yet another option could have been to use BWC07 as $\mathcal{Q}$and create the corresponding $\mathcal{U}$and $\mathcal{R}$. However, this query set is known to suffer from several drawbacks. A new dataset for query segmentation[6] containing manual segment markups collected through crowdsourcing has been recently made publicly available (after we had completed construction of our set) by Hagen et al. [7], but it lacks query-document relevance judgments. These factors motivated us to create a new dataset suitable for our framework, which has been made publicly available (see Sec. 3.4).

5.2 Retrieval using Bing

Bing is a large-scale commercial Web search engine that provides an API service. Instead of Lucene, which is too simplistic, we could have used Bing as the IR engine in our framework. However, such a choice suffers from two drawbacks. First, Bing might be (most probably, it is) already segmenting the query with its own algorithm as a preprocessing step. Second, there is a serious replicability issue. The document pool that Bing uses, i.e. the Web, changes dynamically with documents added and removed from the pool on a regular basis. This makes it difficult to publish a static gold standard dataset with relevance judgments for all appropriate query-URL pairs that the Bing API may retrieve even for the same set of queries. In view of this, the main results were reported in this paper using the Lucene text retrieval system, which can be used to retrieve documents from a static corpus with associated relevance judgments.

However, since we used Bing API to construct $\mathcal{U}$and corresponding $\mathcal{R}$, we have the evaluation statistics using the Bing API as well. For paucity of space, in Table 7 we only present the results for nDCG@10, MRR@10 and MAP@10 for "[12] + Wiki". The table reports results for four quoted version-selection strategies: (i) Unsegmented query only (ii) All segments quoted and (iii) *QVRS*. For all the three metrics, *QVRS* is statistically significantly higher than results for the unsegmented query. Thus, segmentation can play an important role towards improving IR performance of the search engine. We note that the strategy of quoting all the segments is, in fact, detrimental to IR performance. This emphasizes the point that how the segments should be matched in the documents is a very important research challenge. Instead of quoting all the segments, our proposal here is to assume an oracle that will suggest which segments to quote and which are to be left unquoted for the best IR performance. Philosophically, this is a major departure from the previous ideas of using quoted segments, because re-issuing a query by quoting all the segments implies segmentation as a way to generate a fully quoted version of the query (all segments in double quotes). This definition severely limits the scope of segmentation, which ideally should be thought of as a step forward towards better query understanding.

[6]`http://bit.ly/xIhSur`

Table 8: Inter-annotator agreement on features as observed from our experiments.

Feature	Pair 1	Pair 2	Pair 3	Mean
Qry-Acc	0.728	0.644	0.534	0.635
Seg-Prec	0.750	0.732	0.632	0.705
Seg-Rec	0.756	0.775	0.671	0.734
Seg-F	0.753	0.753	0.651	0.719
Seg-Acc	0.911	0.914	0.872	0.899
Rel. judg.	0.962	0.959	0.969	0.963

For relevance judgments, only pairs of (0, 2) and (2, 0) were considered disagreements.

5.3 Inter-annotator agreement

Inter-annotator agreement (IAA) is an important indicator for reliability of manually created data. The IAA for query segmentation is usually around 70% (see, for example, [15]). Table 8 reports the pairwise IAA statistics for H_A, H_B and H_C. Since there are no universally accepted metrics for IAA, we report the values of the five matching metrics when one of the annotations (say H_A) is assumed to be the reference and the remaining pair (H_B and H_C) is evaluated against it. As is evident from the table, the values of all the metrics, except for *Seg-Acc*, is less than 0.78, which indicates a rather low IAA. The value for *Seg-Acc* is close to 0.9, which to the contrary, indicates reasonably high IAA (as in [15]). The last row of Table 8 reports the IAA for the three sets of relevance judgements (therefore, the actual pairs for this column are different from that of the other rows). The agreement in this case is quite high.

There might be several reasons for low IAA for segmentation, such as lack of proper guidelines and/or an inherent disability of human annotators to mark the correct segments of a query. Low IAA raises serious doubts about the reliability of human annotations for query segmentation. On the other hand, high IAA for relevance judgments naturally makes these annotations much more reliable for any evaluation, and strengthens the case for our IR based evaluation framework which only relies on relevance judgments. We note that ideally, relevance judgments should be obtained from the user who has issued the query. This has been referred to as *gold* annotations, as opposed to *silver* or *bronze* annotations which are obtained from expert and non-expert annotators respectively who have not issued the query [1]. Gold annotations are preferable over silver or bronze ones due to relatively higher IAA. Our annotations are *silver standard*, though very high IAA essentially indicates that they might be as reliable as gold standard. The high IAA might be due to the unambiguous nature of the queries.

6. RELATED WORK

Since its inception in 2003 [13], many algorithms have been proposed for automatic segmentation of Web queries. The approaches vary from purely supervised [3] to fully unsupervised [7, 12] machine learning techniques. They differ widely in terms of resources usage (e.g., word n-gram statistics of Web documents [7, 13], Wikipedia titles [15], query logs [12], clickthrough data [9, 10] and various combinations of these) and the underlying algorithmic techniques (e.g., expectation maximization [15] and eigenspace similarity [18]).

6.1 Evaluation on manual annotations

Despite the diversity in approaches to the task, till date there has been only one standard approach for evaluation of query segmentation algorithms, which is to compare the machine output against a set of queries segmented by humans [3, 4, 7, 10, 12, 15, 18]. The basic assumption underlying this evaluation scheme is that humans are capable of segmenting a query in a “correct” or “best possible” way, which, if exploited appropriately, will result in maximum benefits in IR performance. This is probably motivated by the extensive use of human judgments and annotations as the gold standard in the field of NLP (e.g., parts-of-speech labeling, phrase boundary identification, etc.). However, this idea has several shortcomings, as pointed out in Sec. 4.3. Among those who validate query segmentation against human labeled-data, most [3, 4, 6, 7, 10, 15, 18] report accuracies on BWC07, which contains 500 queries selected from the AOL search query log. The popularity of the BWC07 dataset is partly because it was one of the first human annotated datasets created for segmentation, and partly because it is the only publicly available dataset of its kind. While BWC07 has provided a common benchmark for comparing various query segmentation algorithms, there are several limitations of this specific dataset as well as the general approach. BWC07 has only noun phrase queries and there is a non-trivial amount of noise in the annotations. See [7] for a detailed criticism of this dataset.

6.2 IR-based evaluation

There has been only a handful of studies that explore some initial ideas about IR-based evaluation [2, 7, 10] for query segmentation. Bendersky et al. [2] were the first to study the effects of segmentation from an IR perspective. They wanted to see if retrieval quality could be improved by incorporating knowledge of query chunks into an MRF-based retrieval system [11]. Their experiments on different TREC collections using popular IR metrics like MAP indicate that query segmentation can indeed boost IR performance. Li et al. [10] examined the usefulness of query segmentation when built into language models for retrieval, in a Web search setting. However, none of these studies propose an objective IR-based *evaluation framework* for query segmentation. Their scope is limited to the demonstration of one particular strategy for exploiting segmentations for improving IR, instead of evaluating and comparing a set of algorithms.

As an excursus to their main work, Hagen et al. [7] examined if submitting fully quoted queries (generated from algorithm outputs) results in fetching better pages by the search engines. They study the top fifty retrieved documents when the following versions of the queries – unsegmented, manually quoted, quoted by the technique in [3], and by their own method – are submitted to Bing. Assuming the pages retrieved by manual quotation as relevant, it was observed that the technique in [3] achieves the highest average recall. However, the authors also state that such an assumption need not hold good in reality and emphasized the need for an in-depth retrieval-based evaluation.

We would like to emphasize here that the aim of a segmentation technique is not to come up with the best quoted version of a query. While some past works have explicitly or implicitly assumed this definition, there are also other works that view segmentation as a purely structural analysis of a query that identifies chunks or sequences of words that are semantically connected as a unit [10, 12]. By quoting all the segments we would be penalizing the latter philosophy of segmentation which is a more productive and practically useful view of segmentation.

There have been a few studies on detection of noun phrases from queries [5, 19]. This task is similar to query segmentation in the sense that the phrase can be considered as a single unit in the query. Zhang et al. [19] has shown that such phrase detection schemes can actually help in retrieval, and therefore, is along the lines of the philosophy of the present evaluation framework. Nevertheless, as far as we know, this is the first time that a formal conceptual framework for an IR-based evaluation of query segmentation has been proposed. Our study, also for the first time, compares the effectiveness of human segmentation and related matching metrics to an IR-based evaluation.

7. CONCLUSIONS AND FUTURE WORK

End-user of query segmentation is the retrieval engine; hence, it is essential that any segmentation algorithm should be evaluated in an IR-based framework. In this research, we overcome several conceptual challenges to design and implement the first such scheme of evaluation for query segmentation. Using a carefully selected query test set and a group of segmentation strategies, we show that it is possible to have a fair comparison of the relative goodness of each strategy as measured by standard IR metrics. The proposed framework uses resources which are essential for any IR system evaluation, and hence does not require any special input. Our entire dataset – complete with queries, segmentation outputs and relevance judgments – has also been made publicly available to facilitate further research by the community.

Moreover, we gain several useful and non-intuitive insights from the evaluation experiments. Most importantly, we show that human notions of query segments may not be the best for maximizing retrieval performance, and treating them as the gold standard limits the scope for improvement for an algorithm. Also, the matching metrics extensively used till date for comparing against gold standard segmentations can often be misleading. We would like to emphasize that in the future, the focus of IR will mostly shift to tail queries. In such a scenario, an IR-based evaluation scheme gains relevance because validation against a fixed set of gold standard segmentation may often lead to overfitting of the algorithms without yielding any real benefit.

A hypothetical oracle has been shown to be quite useful, but we realize that it will be a much bigger contribution to the community if we could implement a context-aware oracle that can actually tell the search engine which version of a segmented query should be chosen at runtime.

8. REFERENCES

[1] P. Bailey, N. Craswell, I. Soboroff, P. Thomas, A. P. de Vries, and E. Yilmaz. Relevance assessment: are judges exchangeable and does it matter. In *SIGIR '08*, pages 667–674. ACM, 2008.

[2] M. Bendersky, W. B. Croft, and D. A. Smith. Two-stage query segmentation for information retrieval. In *SIGIR '09*, pages 810–811. ACM, 2009.

[3] S. Bergsma and Q. I. Wang. Learning noun phrase query segmentation. In *EMNLP-CoNLL'07*, pages 819–826, 2007.

[4] D. J. Brenes, D. Gayo-Avello, and R. Garcia. On the fly query segmentation using snippets. In *CERI '10*, pages 259–266, 2010.

[5] A. L. da Costa Carvalho, E. S. de Moura, and P. Calado. Using statistical features to find phrasal terms in text collections. *JIDM*, 1(3):583–597, 2010.

[6] M. Hagen, M. Potthast, B. Stein, and C. Bräutigam. The power of naive query segmentation. In *SIGIR '10*, pages 797–798. ACM, 2010.

[7] M. Hagen, M. Potthast, B. Stein, and C. Bräutigam. Query segmentation revisited. In *WWW '11*, pages 97–106, 2011.

[8] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.*, 20:422–446, October 2002.

[9] J. Kiseleva, Q. Guo, E. Agichtein, D. Billsus, and W. Chai. Unsupervised query segmentation using click data: preliminary results. In *WWW '10*, pages 1131–1132. ACM, 2010.

[10] Y. Li, B.-J. P. Hsu, C. Zhai, and K. Wang. Unsupervised query segmentation using clickthrough for information retrieval. In *SIGIR '11*, pages 285–294. ACM, 2011.

[11] D. Metzler and W. B. Croft. A markov random field model for term dependencies. In *SIGIR'05*, pages 472–479, 2005.

[12] N. Mishra, R. Saha Roy, N. Ganguly, S. Laxman, and M. Choudhury. Unsupervised query segmentation using only query logs. In *WWW '11*, pages 91–92. ACM, 2011.

[13] K. M. Risvik, T. Mikolajewski, and P. Boros. Query segmentation for web search. In *WWW (Posters)*, 2003.

[14] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, Inc., 1986.

[15] B. Tan and F. Peng. Unsupervised query segmentation using generative language models and wikipedia. In *WWW '08*, pages 347–356. ACM, 2008.

[16] E. M. Voorhees. Variations in relevance judgments and the measurement of retrieval effectiveness. *Inf. Process. Manage.*, 36:697–716, September 2000.

[17] E. M. Voorhees and D. M. Tice. The trec-8 question answering track evaluation. In *TREC-8*, pages 83–105, 1999.

[18] C. Zhang, N. Sun, X. Hu, T. Huang, and T.-S. Chua. Query segmentation based on eigenspace similarity. In *ACL/AFNLP (Short Papers)'09*, pages 185–188, 2009.

[19] W. Zhang, S. Liu, C. Yu, C. Sun, F. Liu, and W. Meng. Recognition and classification of noun phrases in queries for effective retrieval. In *CIKM '07*, pages 711–720. ACM, 2007.

APPENDIX A: WIKI-BOOST

Algorithm 1 Wiki-Boost(Q', W)

```
 1: $W' \leftarrow \emptyset$
 2: for all $w \in W$ do
 3:     $w' \leftarrow$ Seg-Phase-1($w$)
 4:     $W' \leftarrow W' \cup w'$
 5: end for
 6: $W'$-scores $\leftarrow \emptyset$
 7: for all $w' \in W'$ do
 8:     $w'$-score $\leftarrow PMI(w')$ based on $Q'$
 9:     $W'$-scores $\leftarrow W'$-scores $\cup$ $w'$-score
10: end for
11: U-scores $\leftarrow \emptyset$
12: for all unique unigrams $u \in Q'$ do
13:     u-score $\leftarrow$ probability($u$) in $Q'$
14:     U-scores $\leftarrow$ U-scores $\cup$ u-score
15: end for
16: $W'$-scores $\leftarrow W'$-scores $\cup$ U-scores
17: return $W'$-scores
```

In this appendix, we explain how to augment the output of an n-gram score aggregation based segmentation algorithm with Wikipedia titles[7]. Input to *Wiki-Boost* is a list of queries Q' already segmented by the algorithm in [12] (or any algorithm that meets the above criterion) (say, *Seg-Phase*-1) and W, the list of all stemmed Wikipedia titles ($4,508,386$ entries after removing one-word entries and those with non-ASCII characters). We compute the PMI-score of an n-segment Wikipedia title w' (segmented by *Seg-Phase*-1) by taking the higher of the PMI scores of the first $(n-1)$ segments with the last segment *and* the first segment and the last $(n-1)$ segments. The frequencies of all n-grams are computed from Q'. Scores for unigrams are defined to be their probabilities of occurrence. Thus, the output of the *Wiki-Boost* is a list of PMI-scores for each Wikipedia title in W.

Following this, we use a second segmentation strategy (say, *Seg-Phase*-2) that takes as input q' (the query q segmented by *Seg-Phase*-1) and tries to further join the segments of q' such that the product of scores of the candidate output segments, computed based on the output of *Wiki-Boost*, is maximized. A dynamic programming approach is found to be helpful in searching over all possible segmentations in *Seg-Phase*-2. The output of *Seg-Phase*-2 is the final segmentation output.

[7] `http://dumps.wikimedia.org/enwiki/latest/`, accessed April 6, 2011