Structural Annotation of Search Queries Using Pseudo-Relevance Feedback

Michael Bendersky
bemike@cs.umass.edu

W. Bruce Croft
croft@cs.umass.edu

David A. Smith
dasmith@cs.umass.edu

Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts
Amherst, MA

ABSTRACT

Marking up queries with annotations such as part-of-speech tags, capitalization, and segmentation, is an important part of many approaches to query processing and understanding. Due to their brevity and idiosyncratic structure, search queries pose a challenge to existing annotation tools that are commonly trained on full-length documents. To address this challenge, we view the query as an explicit representation of a latent information need, which allows us to use pseudo-relevance feedback, and to leverage additional information from the document corpus, in order to improve the quality of query annotation.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Algorithms, Experimentation

Keywords

Query processing, query annotation

1. INTRODUCTION

Automatic mark-up of textual corpora with structural annotations is a common practice in natural language processing (NLP) applications, but is done less often in information retrieval (IR). Accordingly, in this paper we focus on the structural annotation of user search queries. There are several key differences between search queries and the corpora usually used in NLP (e.g., news articles or web pages). As previous research shows, these differences severely limit the applicability of standard NLP techniques for annotating query corpora [1, 3, 15]. The most salient difference is length, since search queries are very short. Due to their brevity, the queries often cannot be divided into sub-parts, and do not provide enough context for accurate annotations to be made using the standard NLP tools, which are trained on more syntactically coherent textual units.

Term	Cap	Tag	Seg
where	L	X	B
is	L	VB	I
the	L	X	B
closest	L	X	B
planet	C	NN	B
hollywood	C	NN	I
to	L	X	B
pensacola	C	NN	B
fl	C	NN	I

Table 1: Annotating search query *where is the closest planet hollywood to pensacola fl* with capitalization (L – lowercase, C – otherwise), POS tags (NN – noun, VB – verb, X – otherwise) and segmentation (B/I – beginning of/inside the chunk) mark-up.

However, despite their brevity, queries do differ in both length and grammatical structure [2]. Some queries are keyword concatenations, while others are semi-complete phrases. It is essential for the search engine to correctly interpret the query structure, since it influences both the way the user interacts with the search engine [2] and the way the retrieval should be performed [9]. However, even sentence-like queries are often hard to parse and annotate, as they tend to lack prepositions, proper punctuation, or capitalization, since users (often correctly) assume that these features are disregarded by the retrieval system.

Table 1 presents a simple annotation scheme, exemplified using a web search query. In this scheme, each query is marked-up using three *annotation sequences*: capitalization, POS tags, and segmentation indicators. While this type of simple annotation can be done with a very high accuracy for standard document corpora, it is quite challenging to perform well on queries [1, 3, 11]. Hence, we propose a probabilistic approach that relies on the latent *user information need*, rather than the query itself. This allows us to use *pseudo relevance feedback* (PRF) and leverage the document corpus to improve the quality of query annotation.

The rest of the paper is organized as follows. Sec. 2 details the related work. In Sec. 3 we introduce the query annotation framework. Sec. 4 describes several practical applica-

CIKM'10, October 26–29, 2010, Toronto, Ontario, Canada.
Copyright 2010 ACM 978-1-4503-0099-5/10/10 ...$10.00.

tions of this framework. Sec. 5 presents the experimental results, and Sec. 6 concludes the paper.

2. RELATED WORK

In recent years, structural annotation of search queries has been receiving an increasing attention as an important step toward better query processing and understanding. The literature on query annotation includes query segmentation [3, 12, 9, 20], part-of-speech and semantic tagging [1, 17], named-entity recognition [8, 15, 19, 18], abbreviation disambiguation [21] and stopword detection [14, 11].

In this paper, we advocate the use of pseudo-relevance feedback for query annotation. Pseudo-relevance feedback has proven to be a successful technique for query expansion [6, 13]. Recently, it has also been shown to be effective for query classification [5], query translation [10] and spelling corrections [7].

3. STRUCTURAL ANNOTATION

3.1 Definitions

An explicit representation of the user's latent information need $\mathcal{I}$ is a search query Q, which is a sequence of n terms $q_1, \ldots, q_n$. Given a query Q, our task is to annotate it with the appropriate structure or a set of such structures. We denote an arbitrary structural annotation for Q as Δ_Q.

In this paper, we consider *shallow* structural annotations that can be written as a sequence of annotation symbols $\Delta_Q = [\delta_1, \ldots, \delta_n]$. In other words, each symbol in the annotation sequence corresponds to a single query term.

We define the optimal structural annotation Δ^*_Q for a query Q as the annotation which has the highest probability given the latent information need $\mathcal{I}$ underlying the query:

$$\Delta^*_Q = \operatorname*{argmax}_{\Delta_Q} p(\Delta_Q|\mathcal{I}) \qquad (1)$$

Clearly, the quality of Δ^*_Q will depend on the way the conditional probability $p(\Delta_Q|\mathcal{I})$ is estimated. Next, we describe two ways of estimating this conditional probability.

3.2 Query-based estimation

The most straightforward way to estimate the conditional probability in Eq. 1 is using the explicit representation of the information need $\mathcal{I}$, i.e. the query terms. That is, we can approximate the conditional probability by

$$p(\Delta_Q|\mathcal{I}) \approx p(\Delta_Q|q_1, \ldots, q_n).$$

In practice, to make the resulting estimation feasible, we take a *bag-of-words* approach, and assume independence between both the query terms and the corresponding annotation symbols, Thus, we can rewrite Eq. 1 as:

$$\Delta^{*(QRY)}_Q = \operatorname*{argmax}_{[\delta_1,\ldots,\delta_n]} \prod_{i\in[1,\ldots,n]} p(\delta_i|q_i). \qquad (2)$$

3.3 PRF-based estimation

Given a short, often ungrammatical query, it is hard to accurately estimate the conditional probability in Eq. 1 using the query terms alone. For instance, a keyword query *hawaiian falls*, which refers to a location, will be inaccurately interpreted by a standard POS tagger as a *noun-verb* pair. On the other hand, given a sentence from a corpus that is relevant to the query such as *"Hawaiian Falls is a family-friendly waterpark"*, the word "falls" is correctly identified by a standard POS tagger as a proper noun.

Accordingly, we are interested in bootstrapping the document corpus, in order to better approximate the latent information need $\mathcal{I}$. To this end, we propose employing *pseudo-relevance feedback* — a method that has a long record of success in IR for tasks such as query expansion [6, 13].

In the most general form, given the set of *all* retrievable instances (whole documents or their sub-parts) in the document corpus $\mathcal{C}$, and assuming that Δ_Q and $\mathcal{I}$ are independent given a retrieved instance r, we can derive

$$p(\Delta_Q|\mathcal{I}) = \sum_{r\in\mathcal{C}} p(\Delta_Q|r)p(r|\mathcal{I}).$$

Since for most instances the conditional probability $p(r|\mathcal{I})$ is vanishingly small, we can closely approximate the above by considering only a set of instances R, retrieved at top-k positions in response to Q. This yields

$$p(\Delta_Q|\mathcal{I}) \approx \sum_{r\in R} p(\Delta_Q|r)p(r|\mathcal{I}).$$

Intuitively, the equation above models the information need $\mathcal{I}$ as a mixture of top-k retrieved instances, where each instance is weighted by its relevance to the information need.

Furthermore, to make the estimation of the conditional probability $p(\Delta_Q|r)$ feasible, we assume that the symbols δ_i in the annotation sequence are independent, given an instance r. Note, that this assumption differs from the independence assumption in Eq. 2, since here the annotation symbols are *not independent* given the information need $\mathcal{I}$.

We are now ready to derive the new estimate for Eq. 1, using pseudo-relevance feedback

$$\Delta^{*(PRF)}_Q = \operatorname*{argmax}_{[\delta_1,\ldots,\delta_n]} \sum_{r\in R} \prod_{i\in[1,\ldots,n]} p(\delta_i|r)p(r|\mathcal{I}). \qquad (3)$$

Generally, an estimate of $p(\delta_i|r)$ in this paper will be a smoothed estimator of the form

$$p(\delta_i|r) = \lambda p^{MLE}(\delta_i|r) + (1-\lambda)p^{MLE}(\delta_i|\mathcal{C}),$$

where λ is a constant[1]. This smoothed estimator combines two maximum-likelihood estimators, one based on a retrieved instance r and one based on some large text corpus $\mathcal{C}$. It could be useful in cases when the annotation of the query terms in the top retrieved documents differs significantly from their annotation in the entire collection.

4. APPLICATIONS

In this section we discuss the application of the framework presented in Sec. 3 for practical annotation tasks. For each of these tasks, we define the form that each annotation symbol δ_i in the annotation sequence Δ_Q can take, and the way the conditional probabilities $p(\delta_i|q_i)$ and $p(\delta_i|r)$ (in Eq. 2 and Eq. 3, respectively) are estimated.

4.1 Capitalization

4.1.1 Definition

The capitalization annotation is defined as a sequence $C_Q = [c_1, \ldots, c_n]$ over the query terms $[q_1, \ldots, q_n]$, where

[1] In all the experiments in this paper, $\lambda = 0.8$.

each annotation in the sequence is defined, for simplicity, as a binary trait $c_i \in \{C, L\}$ (capitalized or lowercased)[2]. The task is thus to estimate the conditional probability of a capitalization sequence given an information need — $p(C_Q|\mathcal{I})$.

4.1.2 Estimation

There are two ways to estimate the capitalization of query terms. First, following Eq. 2, we can use only the query terms. In this case, to estimate the conditional probability $p(c_i|q_i)$, we use a maximum likelihood estimator based on the statistics of a large text corpus $\mathcal{C}$

$$p(c_i = x|q_i) \triangleq \frac{\#\{q_i, \mathcal{C}|c_i = x\}}{\#\{q_i, \mathcal{C}\}}, \tag{4}$$

where $x \in \{C, L\}$. Second, we can estimate the capitalization using pseudo-relevance feedback. In this case, to estimate $p(c_i|r)$, we use a smoothed estimate

$$p(c_i = x|r) \triangleq \lambda \frac{\#\{q_i, r|c_i = x\}}{\#\{q_i, r\}} + (1 - \lambda) \frac{\#\{q_i, \mathcal{C}|c_i = x\}}{\#\{q_i, \mathcal{C}\}}. \tag{5}$$

Eq. 5 is a mixture model, which combines the estimate of the capitalization of the query terms in each of the top retrieved instances with the background model of its capitalization in the entire collection.

4.2 POS Tagging

4.2.1 Definition

For this annotation task, we define a simple part-of-speech tagging scheme, consisting of only three tags: nouns, verbs and all other parts-of-speech. While simple, this scheme is quite useful for short queries, where even such simple distinction can be challenging to make. Accordingly the tagging annotation is defined as a sequence $T_Q = [t_1, \ldots, t_n]$ over the query terms, and each annotation in the sequence is defined, as a ternary trait $t_i \in \{NN, VB, X\}$ (noun, verb or other).

4.2.2 Estimation

As in the previous case, there are two ways to estimate POS tagging. The first, following Eq. 2, is to use only the query terms. In this case, to find the optimal tagging $T_Q^{*(QRY)}$, we can simply run an existing POS tagger[3] over the query, and annotate each term using the tagger output (collapsing the actual output tags into the three categories above).

The second way to estimate POS tagging is using pseudo-relevance feedback. Here, to estimate $p(t_w|r)$, we can use a smoothed estimate

$$p(t_i = x|r) \triangleq \lambda \frac{\#\{q_i, r|t_i = x\}}{\#\{q_i, r\}} + (1 - \lambda) p(t_{q_i} = x|q_i) \tag{6}$$

Essentially, the estimate in Eq. 6 leverages the POS tagging of the top-retrieved instances to enhance the initial POS tagging of query terms alone.

4.3 Segmentation

4.3.1 Definition

Segmentation is defined as a sequence of annotations $S_Q = [s_1, \ldots, s_n]$, where each annotation s_i is based on a decision of whether to create a segmentation between terms (q_{i-1}, q_i). Following previous work on query segmentation [3] we define each annotation decision as $s_i \in \{B, I\}$ (beginning of the phrase or inside the phrase).

4.3.2 Estimation

In unsupervised query segmentation, it is common that some term association measure such as mutual information or likelihood ratio is used to determine whether there is a break between two adjacent query terms [3, 12, 20]. Similarly to this prior work, we determine the probability of a segmentation decision $\{B, I\}$ between two terms (q_{i-1}, q_i) based on the strength of association between them in some text x. Hence

$$\hat{p}(s_i = B|q_i, x) = \begin{cases} 1 & \text{if } (assoc_x(q_{i-1}, q_i) \leq \mu) \vee (i = 1) \\ 0 & \text{else,} \end{cases}$$

and

$$\hat{p}(s_i = I|q_i, x) = 1 - \hat{p}(s_i = B|q_i, x),$$

where $assoc_x(q_{i-1}, q_i)$ is a likelihood ratio [16] of terms (q_{i-1}, q_i) occurring together in a text x, and μ is a constant threshold.

The estimate of a probability of an annotation sequence now depends on the way we select the text x, over which the associations between the query terms is computed. There are two possible ways to select x. First, we can simply use the collection $\mathcal{C}$ and get that

$$p(s_i|q_i) = \hat{p}(s_i|q_i, \mathcal{C}), \tag{7}$$

which is similar to the way the unsupervised segmentation of queries is done in the previous work [3, 12, 20].

Alternatively, we can employ pseudo-relevance feedback, using the co-occurrences of query terms in the top retrieved instances. Accordingly, to estimate $p(s_i|r)$, we use a smoothed estimate[4]

$$p(s_i = x|r) = \lambda \hat{p}(s_i = x|q_i, r) + (1 - \lambda) \hat{p}(s_i = x|q_i, \mathcal{C}). \tag{8}$$

5. EVALUATION

5.1 Experimental Setup

For evaluating the performance of our query annotation methods, we sample 250 queries from a search log of a commercial search engine[5], and annotate all of them with three annotations: *capitalization*, *POS tags*, and *segmentation*, according to the description of these annotations in Sec. 4. In this set of 250 queries, there are 96 verbal phrases, 93 questions and 61 keyword queries[6].

For the PRF-based estimates, we use each of these queries to retrieve 50 pages per query using Bing API and index them using Indri[7]. The resulting set of documents constitutes the *document corpus* from which we retrieve sentences

[2] This annotation scheme does not distinguish between acronyms and title-case words (e.g., Mia vs. MIA), however confusions between them are rare, and thus won't be considered here.

[3] http://crftagger.sourceforge.net/

[4] Due to the size differences between the entire collection $\mathcal{C}$ and a single instance r, the threshold μ has to be set separately for each. We set $\mu_{\mathcal{C}} = 10^5$ and $\mu_r = 1$ in all our experiments.

[5] Available as a part of Microsoft 2006 RFP dataset.

[6] Annotations available at http://ciir.cs.umass.edu/~bemike

[7] http://www.lemurproject.org/indri/

CAP	F1 *(% impr)*	MQA *(% impr)*
a-QRY	0.641	0.779
a-PRF	0.711* *(+10.9)*	0.811* *(+4.1)*
TAG	Acc. *(% impr)*	MQA *(% impr)*
a-QRY	0.893	0.878
a-PRF	0.916* *(+2.6)*	0.914* *(+4.1)*
SEG	F1 *(% impr)*	MQA *(% impr)*
a-QRY	0.694	0.672
a-PRF	0.753* *(+8.5)*	0.710* *(+5.7)*

Table 2: Summary of query annotation results. * denotes stat. significant differences with *a-QRY* (two-tailed Fisher's randomization test $\alpha < 0.05$).

	Verbal Phrases		*Questions*		*Keywords*	
CAP	F1	MQA	F1	MQA	F1	MQA
a-QRY	0.621	0.775	0.575	0.843	0.756	0.689
a-PRF	0.750*	0.862*	0.590	0.839	0.784	0.687
TAG	Acc.	MQA	Acc.	MQA	Acc.	MQA
a-QRY	0.894	0.893	0.918	0.918	0.787	0.792
a-PRF	0.908	0.908	0.932	0.935	0.880*	0.890*
SEG	F1	MQA	F1	MQA	F1	MQA
a-QRY	0.705	0.681	0.659	0.644	0.785	0.707
a-PRF	0.751*	0.700	0.740*	0.700*	0.816	0.747

Table 3: Breakdown of query annotation results by query type. * denotes stat. significant differences with *a-QRY* (two-tailed Fisher's randomization test $\alpha < 0.05$).

for constructing set R (see Eq. 3) for the pseudo-relevance feedback estimates. We use Google n-grams [4], as a *background corpus* $\mathcal{C}$.

5.2 Query Annotation

In order to test the effectiveness of the models described in Sec. 4, we compare two annotation methods: *a-QRY* and *a-PRF*. Method *a-QRY* is based on $\Delta_Q^{*(QRY)}$ estimator (Eq. 2). Method *a-PRF* is based on the $\Delta_Q^{*(PRF)}$ estimator (Eq. 3).

For reporting the performance of our methods we use two measures. The first measure treats the annotation decision for each symbol δ_i as a classification. In case of capitalization and segmentation annotations these decisions are binary and we report F1. In case of POS tagging, the decisions are ternary, and hence we report the classification accuracy. We also report an additional measure, MQA (mean query accuracy), which treats the accuracy of the annotation on a query-by-query basis.

In Table 2, we see that *a-PRF* outperforms *a-QRY* for all annotation types, using both performance measures. The improvements are as high as 10.9% for F1 (capitalization), and as high as 5.7% for MQA (segmentation), and, in all cases, are statistically significant. These results verify the performance contributions stemming from using the pseudo-relevance feedback for query annotation.

Table 3 presents a breakdown of the performance of our methods by query type. We note that the contribution of different methods varies significantly across query types. For instance, *a-PRF* has a highly positive impact on POS-tagging estimates for keyword queries. Using *a-PRF* on keyword queries results in 12% improvement over the *a-QRY* baseline, compared to just 4% improvement for all queries.

6. CONCLUSIONS

In this paper, we have investigated two methods for annotating search queries with capitalization, POS tags and segmentation mark-up. First, we examined a query-based method that takes into the account only the query terms and their collection statistics. Second, we proposed a pseudo-relevance feedback based method that takes into the account the top-retrieved instances with response to the query, as well as the query itself.

Our experimental findings over a range of queries from a web search log unequivocally point to the superiority of the PRF-based annotation method over the query-based one. We are encouraged by the success of our PRF-based query annotation technique, and intend to pursue the investigation of its utility for IR applications.

7. ACKNOWLEDGMENT

This work was supported in part by the Center for Intelligent Information Retrieval and in part by ARRA NSF IIS-9014442. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

8. REFERENCES

[1] C. Barr, R. Jones, and M. Regelson. The linguistic structure of english web-search queries. In *Proc. of EMNLP*, 2008.

[2] M. Bendersky and W. B. Croft. Analysis of long queries in a large scale search log. In *Proc. of Workshop on Web Search Click Data*, pages 8–14, 2009.

[3] S. Bergsma and Q. I. Wang. Learning noun phrase query segmentation. In *Proc. of EMNLP*, 2007.

[4] T. Brants and A. Franz. Web 1T 5-gram Version 1, 2006.

[5] A. Z. Broder, M. Fontoura, E. Gabrilovich, A. Joshi, V. Josifovski, and T. Zhang. Robust classification of rare queries using web knowledge. In *Proc. of SIGIR*, pages 231–238, 2007.

[6] C. Buckley. Automatic query expansion using SMART. In *Proc. of TREC-3*, pages 69–80, 1995.

[7] Q. Chen, M. Li, and M. Zhou. Improving query spelling correction using web search results. In *Proc. of EMNLP-CoNLL*, pages 181–189, 2007.

[8] J. Guo, G. Xu, X. Cheng, and H. Li. Named entity recognition in query. In *Proc. of SIGIR*, pages 267–274, 2009.

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

[10] D. He and D. Wu. Enhancing query translation with relevance feedback in translingual information retrieval. *Information Processing and Management*, 2009.

[11] R. Jones and D. C. Fain. Query word deletion prediction. In *Proc. of SIGIR*, pages 435–436, 2003.

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

[13] V. Lavrenko and W. B. Croft. Relevance based language models. In *Proc. of SIGIR*, pages 120–127, 2001.

[14] R. Lo, B. He, and I. Ounis. Automatically building a stopword list for an information retrieval system. In *Proc. of DIR*, 2005.

[15] Y. Lu, F. Peng, G. Mishne, X. Wei, and B. Dumoulin. Improving Web search relevance with semantic features. In *Proceedings of EMNLP*, pages 648–657, 2009.

[16] C. D. Manning and H. Schuetze. *Foundations of Statistical Natural Language Processing*. 1st edition, June 1999.

[17] M. Manshadi and X. Li. Semantic Tagging of Web Search Queries. In *Proc. of ACL*, pages 861–869, 2009.

[18] M. Paşca. Weakly-supervised discovery of named entities using web search queries. In *Proc. of CIKM*, pages 683–690, 2007.

[19] D. Shen, T. Walkery, Z. Zhengy, Q. Yangz, and Y. Li. Personal name classification in web queries. In *Proc. of WSDM*, pages 149–158, 2008.

[20] B. Tan and F. Peng. Unsupervised query segmentation using generative language models and Wikipedia. In *Proceedings of WWW*, pages 347–356, New York, NY, USA, 2008. ACM.

[21] X. Wei, F. Peng, and B. Dumoulin. Analyzing web text association to disambiguate abbreviation in queries. In *Proc. of SIGIR*, pages 751–752, 2008.