Improving Web Search Ranking by Incorporating Structured Annotation of Queries*

Xiao Ding[1], Zhicheng Dou[2], Bing Qin[1], Ting Liu[1], Ji-Rong Wen[3]

[1]Research Center for Social Computing and Information Retrieval
Harbin Institute of Technology, China

[2]Microsoft Research Asia, Beijing 100190, China

[3]Renmin University of China, Beijing, China

[1]{xding, qinb, tliu}@ir.hit.edu.cn;
[2]zhichdou@microsoft.com; [3]jirong.wen@gmail.com

Abstract

Web users are increasingly looking for structured data, such as lyrics, job, or recipes, using unstructured queries on the web. However, retrieving relevant results from such data is a challenging problem due to the unstructured language of the web queries. In this paper, we propose a method to improve web search ranking by detecting *Structured Annotation* of queries based on top search results. In a structured annotation, the original query is split into different units that are associated with semantic attributes in the corresponding domain. We evaluate our techniques using real world queries and achieve significant improvement.

1 Introduction

Search engines are getting more sophisticated by utilizing information from multiple diverse sources. One such valuable source of information is structured and semi-structured data, which is not very difficult to access, owing to information extraction (Wong et al., 2009; Etzioni et al., 2008; Zhai and Liu 2006) and semantic web efforts.

Driving the web search evolution are the user needs. Users usually have a template in mind when formulating queries to search for information. Agarwal et al., (2010) surveyed a search log of 15 million queries from a commercial search engine. They found that 90% of queries follow certain templates. For example, by issuing the query "taylor swift lyrics falling in love", the users are actually seeking for the lyrics of the song "Mary's Song (oh my my my)" by artist Taylor Swift. The words "falling in love" are actually part of the lyrics they are searching for. However, some top search results are irrelevant to the query, although they contain all the query terms. For example, the first top search result shown in Figure 1(a) does not contain the required lyrics. It just contains the lyrics of another song of Taylor Swift, rather than the song that users are seeking.

A possible way to solve the above ranking problem is to understand the underlying query structure. For example, after recognizing that "taylor swift" is an *artist name* and "falling in love" are part of the *lyrics*, we can improve the ranking by comparing the structured query with the corresponding structured data in documents (shown in Figure 1(b)). Some previous studies investigated how to extract structured information from user queries, such as query segmentation (Bergsma and Wang, 2007). The task of query segmentation is to separate the query words into

*Work was done when the first author was visiting Microsoft Research Asia

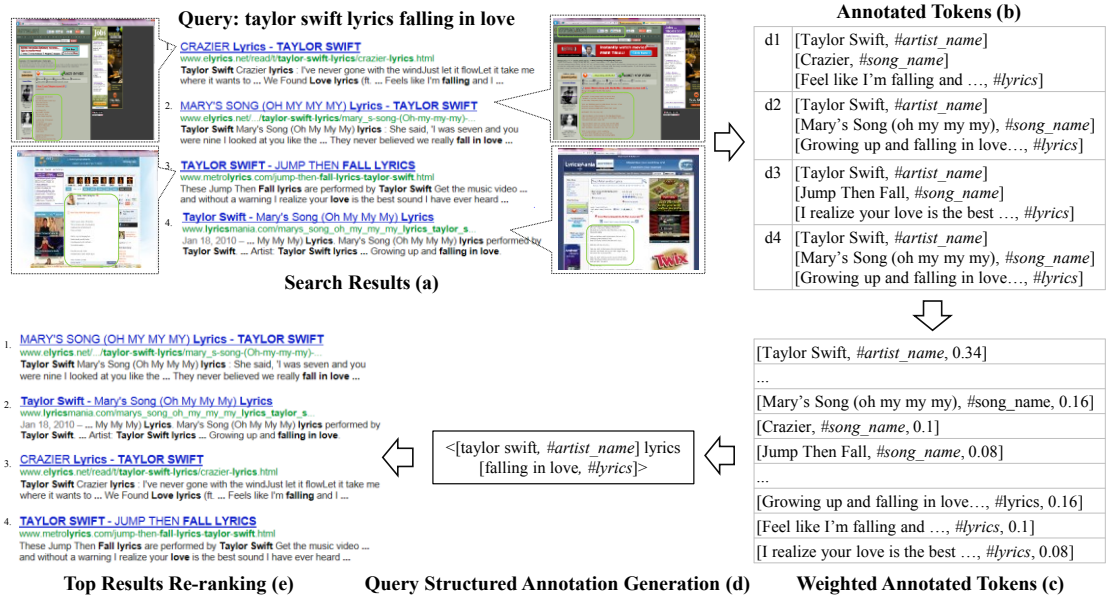


Figure 1. Search results of query "taylor swift lyrics falling in love" and processing pipeline

disjointed segments so that each segment maps to a semantic unit (Li et al., 2011). For example, the segmentation of the query "taylor swift lyrics falling in love" can be "taylor swift | lyrics | falling in love". Since query segmentation cannot tell "talylor swift" is an artist name and "falling in love" are part of lyrics, it is still difficult for us to judge whether each part of the query segmentations matches the right field of the documents or not (such as judge whether "talylor swift" matches the artist name in the document). Recently, a lot of work (Sarkas et al., 2010; Li et al., 2009) proposed the task of structured annotation of queries which aims to detect the structure of the query and assign a specific label to it. However, to our knowledge, the previous methods do not exploit an effective approach for improving web search ranking by incorporating structured annotation of queries.

In this paper, we investigate the possibility of using structured annotation of queries to improve web search ranking. Specifically, we propose a greedy algorithm which uses the structured data (named annotated tokens in Figure 1(b)) extracted from the top search results to annotate the latent structured semantics in web queries. We then compute matching scores between the annotated query and the corresponding structured information contained in documents. The top search results can be re-ranked according to the matching scores. However, it is very difficult to extract structured data from all of the search results. Hence, we propose a relevance feedback based re-ranking model. We use these structured documents whose matching scores are greater than a threshold as feedback documents, to effectively re-rank other search results to bring more relevant and novel information to the user.

Experiments on a large web search dataset from a major commercial search engine show that the F-Measure of structured annotation generated by our approach is as high as 91%. On this dataset, our re-ranking model using the structured annotations significantly outperforms two baselines.

The main contributions of our work include:

1. We propose a novel approach to generate structured annotation of queries based on top search results.
2. Although structured annotation of queries has been studied previously, to the best of our knowledge this is the first paper that attempts to improve web search ranking by incorporating structured annotation of queries.

The rest of this paper is organized as follows. We briefly introduce related work in Section 2. Section 3 presents our method for generating structured annotation of queries. We then propose two novel re-ranking models based on structured annotation in Section 4. Section 5 introduces the data used in this paper. We report experimental results in Section 6. Finally we conclude the work in Section 7.

2 Related Work

There is a great deal of prior research that identifies query structured information. We summarize this research according to their different approaches.

2.1 Structured Annotation of Queries

Recently, a lot of work has been done on understanding query structure (Sarkas et al., 2010; Li et al., 2009; Bendersky et al., 2010). One important method is structured annotation of queries which aims to detect the structure of the query and assign a specific label to it. Li et al., (2009) proposed web query tagging and its goal is to assign to each query term a specified category, roughly corresponding to a list of attributes. A semi-supervised Conditional Random Field (CRF) is used to capture dependencies between query words and to identify the most likely joint assignment of words to "categories." Comparing with previous work, the advantages of our approach are on the following aspects. **First**, we generate structured annotation of queries based on top search results, not some global knowledge base or query logs. **Second**, they mainly focus on the method of generating structured annotation of queries, rather than leverage the generated query structures to improve web search rankings. In this paper, we not only offer a novel solution for generating structured annotation of queries, but also propose a re-ranking approach to improve Web search based on structured annotation of queries. Bendersky et al., (2011) also used top search results to generate structured annotation of queries. However, the annotations in their definition are capitalization, POS tags, and segmentation indicators, which are different from ours.

2.2 Query Template Generation

The concept of query template has been discussed in a few recent papers (Agarwal et al., 2010; Pasca 2011; Liu et al., 2011; Szpektor et al., 2011). A query template is a sequence of terms, where each term could be a word or an attribute. For example, <#artist_name lyrics #lyrics> is a query template, "#artist_name" and "#lyrics" are attributes, and "lyrics" is a word. Structured annotation of queries is different from query template, as a query template can instantiate multiple queries while a structured annotation only serves for a specific query. Unlike query template, our work is ranking-oriented. We aim to automatically annotate query structure based on top search results, and further use these structured annotations to re-rank top search results for improving search performance.

Table 1. Example domain schemas

Domain	Schema	Example structured annotations
lyrics	*#artist_name #song_name #lyrics*	<lyrics of [hey jude, *#song_name*] [beatles, *#artist_name*]>
job	*#category #location*	<[teacher, *#category*] job in [America, *#location*]>
recipe	*#directions #ingredients*	<[baking, # *directions*] [bread, # *ingredients*] recipe>

2.3 Query Segmentation

The task of query segmentation is to separate the query words into disjointed segments so that each segment maps to a semantic unit (Li et al., 2011). Query segmentation techniques have been well studied in recent literature (Tan and Peng, 2008; Yu and Shi, 2009). However, structured annotation of queries cannot only separate the query words into disjoint segments but can also assign each segment a semantic label which can help the search engine to judge whether each part of query segmentation matches the right field of the documents or not.

2.4 Entity Search

The problem of entity search has received a great deal of attention in recent years (Guo et al., 2009; Bron et al., 2010; Cheng et al., 2007). Its goal is to answer information needs that focus on entities. The problem of structured annotation of queries is related to entity search because for some queries, structured annotation items are entities or attributes. Some existing entity search approaches also exploit knowledge from the structure of webpages (Zhao et al., 2005). Annotating query structured information differs from entity search in the following aspects. **First**, structured annotation based ranking is applicable for all queries, rather than just entity related queries. **Second**, the result of an entity search is usually a list of entities, their attributes, and associated homepages, whereas our work uses the structured information from webpages to annotate query structured information and further leverage structured annotation of queries to re-rank top search results.

3 Structured Annotation of Queries

3.1 Problem Definition

We start our discussion by defining some basic concepts. A *token* is defined as a sequence of words including space, i.e., one or more words. For example, the bigram "taylor swift" can be a single token. As our objective is to find structured annotation of queries in a specific domain, we begin with a definition of domain schema.

Definition 1 (Domain Schema): For a given domain of interest, the domain schema is the set of attributes. We denote the domain schema as $A = \{a_1, a_2, \cdots, a_n\}$, where each a_i is the name of an *attribute* of the domain. Sample domain schemas are shown in Table 1. In contrast to previous methods (Agarwal et al., 2010), our definition of domain schema does not need attribute values. For the sake of simplicity, this paper assumes that attributes in domain schema are available. However, it is not difficult to pre-specify attributes in a specific domain.

Definition 2 (Annotated Token): An *annotated token* in a specific domain is a pair $[v, a]$, where v is a token and a is a corresponding *attribute* for v in this domain. [hey jude, *#song_name*] is an example of an annotated token for the "lyrics" domain shown in Table 1. The words "hey jude" comprise a token, and its corresponding attribute name is *#song_name*. If a token does not have any corresponding attributes, we denote it as *free token*.

Definition 3 (Structured Annotation): A *structured annotation p* is a sequence of terms $< s_1, s_2, \cdots, s_k >$, where each s_i could be a free token or an annotated token, and at least one of the terms is an annotated token, i.e., $\exists i \in [1, k]$ for which s_i is an annotated token.

Given the schema for the domain "lyrics", <[taylor swift, *#artist_name*] lyrics [falling in love, *#lyrics*]> is a possible structured annotation for the query "taylor swift lyrics falling in love". In this annotation, [taylor swift, *#artist_name*] and [falling in love, *#lyrics*] are two annotated tokens. The word "lyrics" is a free token.

Intuitively, a structured annotation corresponds to an interpretation of the query as a request for some structured information from documents. The set of annotated tokens expresses the information need of the documents that have been requested. The free tokens may provide more diverse information. Annotated tokens and free tokens together cover all query terms, reflecting the complete user intent of the query.

Algorithm 1: Query Structured Annotation Generation

Input: a list of weighted annotated tokens $T = \{t_1, \ldots, t_m\}$;
a query $q =$ "$w_1, \ldots, w_n$" where $w_i \in$ W;
a pre-defined threshold score δ.
Output: a query structured annotation $p = <s_1, \ldots, s_k>$.

1: Set $p = q = \{s_1, \ldots, s_n\}$, where $s_i = w_i$
2: for u = 1 to *T.size* do
3: compute $Match(p, t_u)$
$= Match(p, t_u.v)$
$= t_u.w \times max_{0 \le i < j \le n} Sim(p_{ij}, t_u.v)$,
where $p_{ij} = s_i, \ldots, s_j$, s.t. $s_l \in$ W for $l \in [i, j]$. *//p_{ij} is just in the remaining query words*
4: end for
5: find the maximum matching t_u with
$t_{max} = argmax_{1 \le u \le m} Match(p, t_u)$
6: if $Match(p, t_{max}) > \delta$ then
7: replace $s_i, \ldots, s_j$ in p with $[s_i, \ldots, s_j, t_{max}.a]$
8: remove t_{max} from T
9: n ← n – (j - i)
10: go to step 2
11: else
12: return p
13: end if

3.2 Generating Structured Annotation

In this paper, given a domain schema A, we generate structured annotation for a query q based on the top search results of q. We propose using top search results, rather than some global knowledge base or query logs, because:

(1) Top search results have been proven to be a successful technique for query explanation (Bendersky et al., 2010).

(2) We have observed that in most cases, a reasonable percentage of the top search results are relevant to the query. By aggregating structured information from the top search results, we can get more query-dependent annotated tokens than using global data sources which may contain more noise and outdated.

(3) Our goal for generating structured annotation is to improve the ranking quality of queries. Using top search results enables simultaneous and consistent detection of structured information from documents and queries.

As mentioned in Section 3.1, we generate structured annotation of queries based on annotated tokens, which are actually structured data (shown in Figure 1(b)) embedded in web documents. In this paper, we assume that the annotated tokens are

available and we mainly focus on how to use these annotated tokens from top search results to generate structured annotation of queries. The approach is comprised of two parts, one for weighting annotated tokens and the other for generating structured annotation of queries based on the weighted annotated tokens.

Weighting: As shown in Figure 1, annotated tokens extracted from top results may be inconsistent, and hence some of the extracted annotated tokens are less useful or even useless for generating structured annotation.

We assume that a better annotated token should be supported by more top results; while a worse annotated token may appear in fewer results. Hence we aggregate all the annotated tokens extracted from top search results, and evaluate the importance of each unique one by a ranking-aware voting model as follows. For an annotated token [*v*, *a*], its weight *w* is defined as:

$$w = \frac{1}{N}\sum_{1\le j\le N} w_j \quad (1)$$

where w_j is a voting from document d_j, and

$$w_j = \begin{cases} \frac{N-j+1}{N}, & \text{if } [v,a] \in d_j \\ 0, & \text{else} \end{cases}$$

Here, *N* is the number of top search results and *j* is the ranking position of document d_j. We then generate a weighted annotated token [*v*, *a*, *w*] for each original unique token [*v*, *a*].

Generating: The process by which we map a query *q* to *Structured Annotation* is shown in Algorithm 1. The algorithm takes as input a list of weighted annotated tokens and the query *q*, and outputs the structured annotation of the query *q*. The algorithm first partitions the query *q* by comparing each sub-sequence of the query with all the weighted annotated tokens, and find the maximum matching annotated token (line 1 to line 5). Then, if the degree of match is greater than the threshold δ which is a pre-defined threshold score for fuzzy string matching, the query substring will be assigned the attribute label of the maximum matching annotated token (line 6 to line 8). The algorithm stops when all the weighted annotated tokens have been scanned, and outputs the structured annotation of the query.

Note that in some cases, the query may fail to exactly match with the annotated tokens, due to spelling errors, acronyms or abbreviations in users' queries. For example, in the query "broken and **beatuful** lyrics", "broken and **beatuful**" is a misspelling of "broken and **beautiful**." We adopt a fuzzy string matching function for comparing a sub-sequence string *s* with a token *v*:

$$Sim(s,v) = 1 - \frac{EditDistance(s,v)}{\max(|s|,|v|)} \quad (2)$$

where *EditDistance*(*s*, *v*) measures the edit distances of two strings, |*s*| is the length of string *s* and |*v*| is the length of string *v*.

4 Ranking with Structured Annotation

Given a domain schema $A = \{a_1, a_2, \cdots, a_n\}$, and a query *q*, suppose that $p = < s_1, s_2, \ldots, s_k >$ is the structured annotation for query *q* obtained using the method introduced in the above sections. *p* can better reflects the user's real search intent than the original *q*, as it presents the structured semantic information needed instead of a simple word string. Therefore, a document d_i can better satisfy a user's information need if it contains corresponding structured semantic information in *p*. Suppose that T_i is the set of annotated tokens extracted from document d_i, we compute a re-ranking score, denoted by *RScore*, for document d_i as follows:

$$\begin{aligned} RScore(q, d_i) &= Match(q, d_i) \\ &= Match(p, T_i) \\ &= \sum_{1\le j\le k}\sum_{t\in T_i} Match(s_j, t) \end{aligned}$$

where

$$Match(s_j, t) = \begin{cases} Sim(s_j.v_j, t.v), & \text{if } s_j.a_j = t.a \\ 0, & \text{else} \end{cases} \quad (3)$$

where s_j is an annotated token in *p* and *t* is an annotated token in d_i. We use Equation (2) to compute the similarity between values in query annotated tokens and values in document annotated tokens. We propose two re-ranking models, namely the conservative re-ranking model, to re-rank top results based on *RScore* and relevance feedback based re-ranking model.

4.1 Conservative Re-ranking Model

A nature way to re-rank top search results is according to their *RScore*. However, we fail to obtain annotated tokens from some retrieved documents, and hence the *RScore* of these documents are not available. In the conservative re-ranking model, we only re-rank search results that have an *RScore*. For example, suppose there are five retrieved documents {d_1, d_2, d_3, d_4, d_5} for query *q*, we can extract structured information from document d_3 and d_4 and $RScore(q, d_4) > RScore(q, d_3)$. Note that we cannot obtain

structured information from d_1, d_2, and d_5. In the conservative re-ranking method, d_1, d_2, and d_5 retain their original positions; while d_3 and d_4 will be re-ranked according to their *RScore*. Therefore, the final ranking generated by our conservative re-ranking model should be $\{d_1, d_2, d_4, d_3, d_5\}$, in which the documents are re-ranked among the affected positions.

There is also useful information in the documents without structured data, such as community question answering websites. However, in the conservative re-ranking model they will not be re-ranked. This may hurt the performance of our re-ranking model. One reasonable solution is relevance feedback model.

4.2 Relevance Feedback based Re-ranking Model

The disadvantage of the conservative re-ranking model is that it only can re-rank those top search results with structured data. To make up its limitation, we propose a relevance feedback based re-ranking model. The key idea of this model is based on the observation that the search results with the corrected annotated tokens could give implicit feedback information. Hence, we use these structured documents whose *RScore* are greater than a threshold γ (empirically set it as 0.6) as feedback documents, to effectively re-rank other search results to bring more relevant and novel information to the user.

Formally, given a query Q and a document collection C, a retrieval system returns a ranked list of documents D. Let d_i denote the i-th ranked document in the ranked list. Our goal is to study how to use these feedback documents, $J \subseteq \{d_1, \ldots, d_k\}$, to effectively re-rank the other r search results: $U \subseteq \{d_{k+1}, \ldots, d_{k+r}\}$. A general formula of relevance feedback model (Salton et al, 1990) R is as follows:

$$R(Q') = (1 - \alpha)L_q(\mathrm{Q}) + \alpha L_d(\mathrm{J}) \qquad (4)$$

where $\alpha \in [0, 1]$ is the feedback coefficient, and L_q and L_d are two models that map a query and a set of relevant documents, respectively, into some comparable representations. For example, they can be represented as vectors of weighted terms or language models.

In this paper, we explore the problem in the language model framework, particularly the KL-divergence retrieval model and mixture-model feedback method (Zhai and Lafferty, 2001), mainly because language models deliver state-of-the-art retrieval performance and the mixture-model based feedback is one of the most effective feedback techniques which outperforms Rocchio feedback.

4.2.1 The KL-Divergence Retrieval Model

The KL-divergence retrieval model was introduced in Lafferty and Zhai, (2001) as a special case of the risk minimization retrieval framework and can support feedback more naturally. In this model, queries and documents are represented by unigram language models. Assuming that these language models can be appropriately estimated, KL-divergence retrieval model measures the relevance value of a document D with respect to a query Q by computing the negative Kullback-Leibler divergence between the query language model θ_Q and the document language model θ_D as follows:

$$S(Q, D) = -D(\theta_Q || \theta_D) = -\sum_{w \in V} p(w|\theta_Q) log \frac{p(w|\theta_Q)}{p(w|\theta_D)} \qquad (5)$$

where V is the set of words in our vocabulary. Intuitively, the retrieval performance of the KL-divergence relies on the estimation of the document model θ_D and the query model θ_Q.

For the set of k relevant documents, the document model θ_D is estimated as $p(\mathrm{w}|\theta_D) = \frac{1}{k}\sum_{i=1}^{k} \frac{c(w,r_i)}{|r_i|}$, where $c(w, r_i)$ is the count of word w in the i-th relevant document, and $|r_i|$ is the total number of words in that document. The document model θ_D needs to be smoothed and an effective method is Dirichlet smoothing (Zhai et al., 2001).

The query model intuitively captures what the user is interested in, and thus would affect retrieval performance. With feedback documents, θ_Q is estimated by the mixture-model feedback method.

4.2.2 The Mixture Model Feedback Method

As the problem definition in Equation (4), the query model can be estimated by the original query model $p(w|\theta_Q) = \frac{c(w,Q)}{|Q|}$ (where $c(w,Q)$ is the count of word w in the query Q, and $|Q|$ is the total number of words in the query) and the feedback document model. Zhai and Lafferty, (2001) proposed a mixture model feedback method to estimate the feedback document model. More specifically, the model assumes that the feedback documents can be generated by a background language model $p(w|C)$ estimated using the whole collection and an unknown topic language model

Table 2. Domain queries used in our experiment

Domain	Containing Keyword	Queries	Structured Annotation%
lyrics	"lyrics"	196	95%
job	"job"	124	92%
recipe	"recipe"	76	93%

Table 3. Quality of Structured Annotation. All the improvements are significant ($p < 0.05$)

Domain	Method	Precision	Recall	F-Measure
lyrics	Baseline	90.06%	84.92%	87.41%
	Our	95.45%	89.83%	92.55%
job	Baseline	89.62%	80.14%	84.62%
	Our	95.31%	84.93%	89.82%
recipe	Baseline	83.96%	84.23%	84.09%
	Our	89.68%	88.44%	89.06%
All	Baseline	87.88%	83.10%	85.42%
	Our	**93.61%**	**88.45%**	**90.96%**

θ_F to be estimated. Formally, let $F \subset C$ be a set of feedback documents. In this paper, F is comprised of documents that *RScore* are greater than γ. The log-likelihood function of the mixture model is:

$$log(F|\theta_F) = \sum_{D \in F} \sum_{w \in V} c(w, D) \log[(1-\lambda)p(w|\theta_F) + \lambda p(w|C)] \quad (6)$$

where $\lambda \in [0,1)$ is a mixture noise parameter which controls the weight of the background model. Given a fixed λ, a standard EM algorithm can then be used to estimate $p(w|\theta_F)$, which is then interpolated with the original query model $p(w|Q)$ to obtain an improved estimation of the query model:

$$p(w|\theta_Q) = (1-\alpha)p(w|Q) + \alpha p(w|\theta_F) \quad (7)$$

where α is the feedback coefficient.

5 Data

We used a dataset composed of 12,396 queries randomly sampled from query logs of a search engine. For each query, we retrieved its top 100 results from a commercial search engine. The documents were judged by human editors. A five-grade (from 0 to 4 meaning from bad to perfect) relevance rating was assigned for each document.

We used a proprietary query domain classifier to identify queries in three domains, namely "lyrics," "recipe," and "job," from the dataset. The statistics about these domains are shown in Table 2. To investigate how many queries may potentially have structured annotations, we manually created structured annotations for these queries. The last column of Table 2 shows the percentage of queries

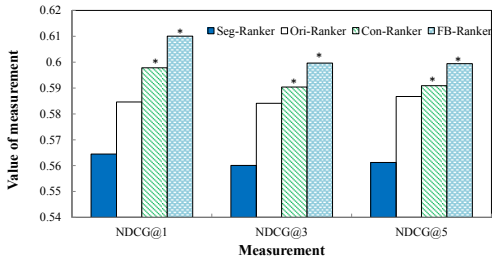


Figure 2. Ranking Quality (* indicates significant improvement)

that have structured annotations created by annotators. We found that for each domain, there was on average more than 90% of queries identified by us that had a certain structured annotation. This indicates that a large percentage of these queries contain structured information, as we expected.

6 Experimental Results

In this section, we present the structured annotation of queries and further re-rank the top search results for the three domains introduced in Section 5. We used the ranking returned by a commercial search engine as our one of the **Baselines**. Note that as the baseline already uses a large number of ranking signals, it is very difficult to improve it any further.

We evaluate the ranking quality using the widely used Normalized Discounted Cumulative Gain measure (NDCG) (Javelin and Kekalainen., 2000). We use the same configuration for NDCG as (Burges et al. 2005). More specifically, for a given query q, the NDCG@K is computed as:

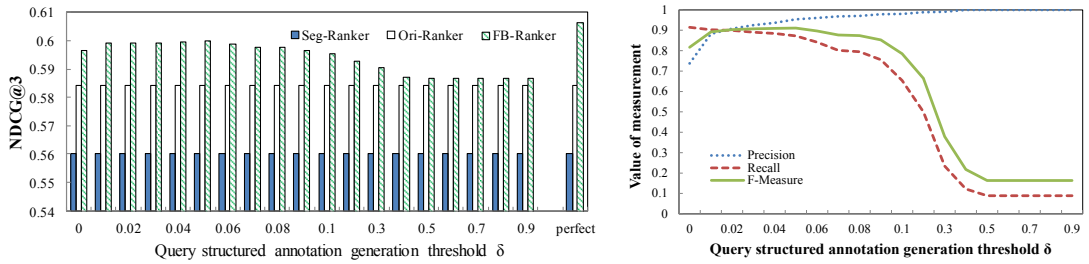
$$N_q = \frac{1}{M_q} \frac{\sum_{j=1}^{K} (2^{r(j)} - 1)}{\log(1+j)} \quad (4)$$

M_q is a normalization constant (the ideal NDCG) so that a perfect ordering would obtain an NDCG of 1; and $r(j)$ is the rating score of the *j-th* document in the ranking list.

6.1 Overall Results

6.1.1 Quality of Structured Annotation of Queries

We generated the structured annotation of queries based on the top 10 search results and used $\delta = 0.04$ for Algorithm 1. We used several existing metrics, P (Precision), R (Recall), and F-Measure to evaluate the quality of the structured annotation. As a query structured annotation may contain more than one annotated token, we concluded that the



(a) Quality of re-ranking (b) Quality of query structured annotation

Figure 3. Quality of re-ranking and quality of query structured annotation with different number of search results

Table 4. Detailed ranking results on three domains. All the improvements are significant ($p < 0.05$)

Domain	Ranking Method	NDCG@1	NDCG@3	NDCG@5
lyrics	Seg-Ranker	0.572	0.574	0.575
	Ori-Ranker	0.621	0.628	0.636
	FB-Ranker	**0.637**	**0.639**	**0.647**
recipe	Seg-Ranker	0.629	0.631	0.634
	Ori-Ranker	0.678	0.687	0.696
	FB-Ranker	**0.707**	**0.704**	**0.709**
job	Seg-Ranker	0.438	0.413	0.408
	Ori-Ranker	0.470	0.453	0.442
	FB-Ranker	**0.504**	**0.474**	**0.459**

annotation was correct only if the entire annotation was **completely the same** as the annotation labeled by annotators. Otherwise we treated the structured annotation as incorrect. Experimental results for the three domains are shown in Table 3. We compare our approach with Xiao Li, (2010) (denoted as baseline), on the dataset described in Section 5. They labeled the semantic structure of noun phrase queries based on semi-Markov CRFs. Our approach achieves better performance than the baseline (about 5.5% significant improvement on F-Measure). This indicates that the approach of generating structured annotation based on the top search results is more effective. With the high-quality structured annotation of queries in hand, it may be possible to obtain better ranking results using our proposed re-ranking models.

6.1.2 Re-ranking Result

We used the models introduced in Section 4 to re-rank the top 10 search results, based on structured annotation of queries and annotated tokens.

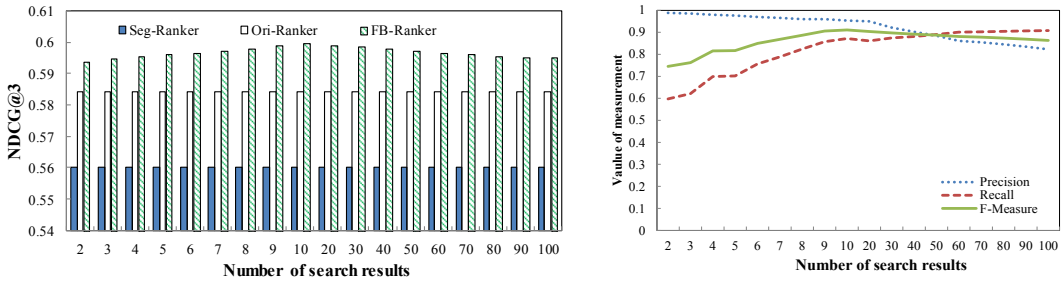
Recall that our goal is to quantify the effectiveness of structured annotation of queries for real web search. One dimension is to compare with the original search results of a commercial search engine (denoted as **Ori-Ranker**). The other is to compare with the query segmentation based re-ranking model (denoted as **Seg-Ranker**; Li et al., 2011) which tries to improve web search ranking by incorporating query segmentation. Li et al., (2011) incorporated query segmentation in the BM25, unigram language model and bigram language model retrieval framework, and bigram language model achieved the best performance. In this paper, Seg-Ranker integrates bigram language model with query segmentation.

The ranking results of these models are shown in Figure 2. This figure shows that all our two rankers significantly outperform the Ori-Ranker– the original search results of a commercial search engine. This means that using high-quality structured annotation does help better understanding of user intent. By comparing these structured annotations and the annotated tokens in documents, we can re-rank the more relevant results higher and yield better ranking quality.

Figure 2 also suggests that structured annotation based re-ranking models outperform query segmentation based re-ranking model. This is mainly because structured annotation can not only separate the query words into disjoint segments but can also assign each segment a semantic label. Taking full advantage of the semantic label can lead to better ranking performance.

Furthermore, Figure 2 shows that FB-Ranker outperforms Con-Ranker. The main reason is that in Con-Ranker, we can only reasonably re-rank the search results with structured data. However, in FB-Ranker we can not only re-rank the structured search results but also can re-rank other documents by incorporating implicit information from those structured documents.

On average, FB-Ranker achieves the best ranking performance. Table 4 shows more detailed



(a) Quality of re-ranking (b) Quality of query structured annotation

Figure 4. Quality of re-ranking and quality of query structured annotation with different number of search results

results for the three selected domains. This table shows that FB-Ranker consistently outperforms the two baseline rankers on these domains. In the remaining part of this paper, we will only report the results for this ranker, due to space limitations. Table 4 also indicates that we can get robust ranking improvement in different domains, and we will consider applying it to more domains.

6.2 Experiment with Different Thresholds of Query Structured Annotation Algorithm

As introduced in Algorithm 1, we pre-defined a threshold δ for fuzzy string matching. We evaluated the quality of re-ranking and query structured annotation with different settings for δ. The results are shown in Figure 3. We found that:

(1) When we use $\delta = 0$, which means that the structured annotations can be generated no matter how small the similarity between the query string and a weighted annotated token is, we can get a significant NDCG@3 gain of 2.15%. Figure 3(b) shows that the precision of the structured annotation is lowest when $\delta = 0$. However, the precision is still as high as 0.7375, and the highest recall is obtained in this case. This means that the quality of the generated structured annotations is still reasonable, and hence we can get a ranking improvement when $\delta = 0$, as shown in Figure 3(a).

(2) Figure 3(a) suggests that the quality of re-ranking increases when the threshold δ increases from 0 to 0.05. It then decreases when δ increases from 0.06 to 0.5. Comparing these two figures shows that the trend of re-ranking performance adheres to the quality of the structured annotation. The settings for δ dramatically affect the recall and precision of the structured annotation; and hence the ranking quality is impacted. The larger δ is, the lower the recall of the structured annotation is.

(3) Since the re-ranking performance dramatically changes along with the quality of the structured annotation, we conducted a re-ranking experiment with perfect structured annotations (F-Measure equal to 1.0). Perfect structured annotations mean the annotations created by annotators as introduced in Section 5. The results are shown in the last bar of Figure 3(a). We did not find a large space for ranking improvement. The NDCG@3 when using perfect structured annotations was 0.606, which is just slightly better than our best result (yield when δ=0.05). It indicates that our structured annotation generation algorithm is already quite effective.

(4) Figure 3(a) shows that our approach outperforms the two baseline approaches with most settings for δ. This indicates that our approach is relatively stable with different settings for δ.

6.3 Experiment with Number of Top Search Results

The above experiments are conducted based on the top 10 search results. In this section, by adjusting the number of top search results, ranging from 2 to 100, we investigate whether the quality of structured annotation of queries and the performance of re-ranking are affected by the quantity of search results. The results shown in Figure 4 indicate that the number of search results does affect the quality of structured annotation of queries and the performance of re-ranking. Structured annotations of queries become better when more search results are used from 2 to 20. This is because more search results cover more websites in our domain list, and hence can generate more annotated tokens. More results also provide more evidence for voting the importance of

annotated tokens, and hence can improve the quality of structured annotation of queries.

In addition, we also found that structured annotation of queries become worse when too many lower ranked results are used (e.g, using results ranked lower than 20). This is because the lower ranked results are less relevant than the higher ranked results. They may contain more irrelevant or noisy annotated tokens than higher ranked documents; and hence using them may harm the precision of the structured annotations. Figure 4 also indicates that the quality of ranking and the accuracy of structured annotations are correlated.

7 Conclusions

In this paper, we studied the problem of improving web search ranking by incorporating structured annotation of queries. We proposed a systematic solution, first to generate structured annotation of queries based on top search results, and then launching two structured annotation based re-ranking models. We performed a large-scale evaluation over 12,396 queries from a major search engine. The experiment results show that the F-Measure of query structured annotation generated by our approach is as high as 91%. In the same dataset, our structured annotation based re-ranking model significantly outperforms the original ranker – the ranking of a major search engine, with improvements 5.2%.

Acknowledgments

This work was supported by National Natural Science Foundation of China (NSFC) via grant 61273321, 61133012 and the Nation-al 863 Leading Technology Research Project via grant 2012AA011102.

References

G. Agarwal, G. Kabra, and K. C.-C. Chang. Towards rich query interpretation: walking back and forth for mining query templates. In *Proc. of WWW '10*.

M. Bendersky, W. Bruce Croft and D. A. Smith. Joint Annotation of Search Queries, In *Proc. of ACL-HLT 2011*.

M. Bendersky, W. Bruce Croft and D. A. Smith. Structural Annotation of Search Queries Using Pseudo-Relevance Feedback, In *Proc. Of CIKM 2010.*

S. Bergsma and Q. I. Wang. Learning noun phrase query segmentation. In *Proceedings of EMNLP-CoNLL'07*.

M. Bron, K. Balog, and M. de Rijke. Ranking related entities: components and analyses. In *Proc. of CIKM '10*.

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

C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank usinggradient descent. In *Proceedings of ICML '05*.

T. Cheng, X. Yan, and K. C.-C. Chang. Supporting entity search: a large-scale prototype search engine. In *Proc. of SIGMOD '07*.

O. Etzioni, M. Banko, S. Soderland, and D.S. Weld, (2008). Open Information Extraction from the Web, *Communications of the ACM*, 51(12): 68-74.

J. Guo, G. Xu, X. Cheng, and H. Li. Named entity recognition in query. In *Proc. Of SIGIR' 2009.*

K. Jarvelin and J. Kekalainen. Ir evaluation methods for retrieving highly relevant documents. In *SIGIR '00*.

J. Lafferty and C. Zhai, Document language models, query models, and risk minimization for information retrieval, In *Proceedings of SIGIR'01*, pages 111-119, 2001.

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

Y. Li, BJP. Hsu, CX. Zhai and K. Wang. Unsupervised Query Segmentation Using Clickthrough for Information Retrieval. In *Proc. of SIGIR'11*.

X. Li, Y.-Y. Wang, and A. Acero. Extracting structured information from user queries with semi-supervised conditional random fields. In *Proc. of SIGIR'09*.

Y. Liu, X. Ni, J-T. Sun, Z. Chen. Unsupervised Transactional Query Classification Based on Webpage Form Understanding. In *Proc. of* CIKM *'11*.

Y. Liu, M. Zhang, L. Ru, and S. Ma. Automatic query type identification based on click-through information. In *LNCS*, 2006.

M. Pasca. Asking What No One Has Asked Before: Using Phrase Similarities To Generate Synthetic Web Search Queries. In *Proc. of CIKM '11*.

G. Salton and C. Buckley. Improving retrieval performance by relevance feedback. Journal of the American Society for Information Science, 41(4):288-297, 1990.

N. Sarkas, S. Paparizos, and P. Tsaparas. Structured annotations of web queries. In *Proc. of SIGMOD'*10.

I. Szpektor, A. Gionis, and Y. Maarek. Improving recommendation for long-tail queries via templates. In *Proc. of WWW '11*

B. Tan and F. Peng. Unsupervised query segmentation using generative language models and wikipedia. In WWW'08.

T.-L. Wong, W. Lam, and B. Chen. Mining employment market via text block detection and adaptive cross-domain information extraction. In *Proc. SIGIR*, pages 283–290, 2009.

X. Yu and H. Shi. Query segmentation using conditional random fields. In *Proceedings of* KEYS *'*09.

C. Zhai and J. Lafferty, Model-based feedback in the language modeling approach to information retrieval , In *Proceedings of CIKM'01*, pages 403-410, 2001.

C. Zhai and J. Lafferty, A study of smoothing methods for language models applied to ad hoc information retrieval, In *Proceedings of SIGIR'01*, pages 334-342, 2001.

Y. Zhai and B. Liu. Structured data extraction from the Web based on partial tree alignment. *IEEE Trans. Knowl. Data Eng.*, 18(12):1614−1628, Dec. 2006.

H. Zhao, W. Meng, Z. Wu, V. Raghavan, and C. Yu. Fully automatic wrapper generation for search engines. In *Proceedings of WWW '05.*

S. Zheng, R. Song, J.-R. Wen, and D. Wu. Joint optimization of wrapper generation and template detection. In *Proc. of SIGKDD'*07.