



TECHNICAL REPORT

YL-2009-007

**QUERYTEXT - USING QUERIES AND CLICKS TO IMPROVE TEXT MATCHING
FOR WEB SEARCH**

Hang Cui, Srihari Reddy, and Donald Metzler
Yahoo! Labs
{cui, sriharir, metzler}@yahoo-inc.com

Bangalore • Barcelona • Haifa • Montreal • New York
Santiago • Silicon Valley

Yahoo! Labs Technical Report No. YL-2009-007

**QUERYTEXT – USING QUERIES AND CLICKS TO IMPROVE TEXT MATCHING
FOR WEB SEARCH**

Hang Cui, Srihari Reddy, and Donald Metzler

Yahoo! Labs

{cui, sriharir, metzler}@yahoo-inc.com

ABSTRACT: User queries and their associated clicks have been extensively explored to improve Web search relevance. Very little existing work explores how user clicks can be used to improve text matching for Web search. In this paper, we treat user queries that result in clicks as off-page annotations. These queries, like anchor text, provide a valuable additional source of relevance information for Web pages. We call the queries that are used to annotate Web pages in this way the QueryText. We propose using the QueryText as a new weighted textual field for Web pages, where the weights are based on user click behavior. We derive two sets of text matching features from the new field – BM25F-based features and n -gram features. We implement the features within a commercial search engine and evaluate the effectiveness of our approach on real large-scale Web data. Our evaluation results show significant improvements in retrieval effectiveness using text match features derived from the QueryText.

1. Introduction

Recent years have witnessed the prosperous growth of Web search. Millions of users rely on the Web to obtain necessary information. Search engines act as a bridge to connect the information needs of users to the information available on the Web. Web search is difficult due to its dynamic nature – both Web pages and user’s information needs are changing rapidly. One common problem for Web search is how to represent Web pages (also called documents in this paper) to enable search engines to satisfy users’ information needs better. It is well known that there is a lexical gap between user queries and Web page content. Web pages are represented as structured (fielded) documents. In addition to traditional fields like title and body, anchor text is also extensively used when representing Web pages [4]. Search engines measure the similarity of user queries against the anchor text, title, body, and other fields to compute a textual similarity score for the document. This score is typically combined with other features, such as spam and link analysis features to effectively rank documents.

Anchor text provides a valuable source of information for Web search, especially for computing textual similarity scores. However, anchor text may only partially address the problem of bridging the lexical gap between queries and Web pages. This is the case because anchor text is created by Web page publishers and may not be updated very frequently. For example, common pieces of anchor text include “click here” and “homepage”, which are more navigational in nature and do not describe the topical content of a Web page. Therefore, it is desirable to find other sources of textual information, above and beyond anchor text, that better represents Web page content.

Users of web search engines provide implicit relevance feedback in the form of clicks during their daily search sessions. With the accumulation of user queries and other search behavior, user search logs have become another source for capturing user intent. User search logs record various user behavior during their search sessions, including the queries issued, results returned, and the results clicked by the user. This paper explores how information in search logs can be used as another textual source for Web pages. When a user clicks on a web page, we associate the query issued with the clicked page. Furthermore, the collective user behavior, in the form of aggregate click statistics, can be utilized to give prior importance (or weights) to the queries associated with pages. The rationale is that users do not randomly click on pages, especially given that search result presentation in current search engines has been greatly improved by using title, URL and summary with highlighted search keywords. This has been demonstrated by previous work that utilizes user clicks as implicit feedback for result relevance [1].

In this paper, we propose using user queries as a new *weighted* textual field for representing Web pages. First, for each Web page, we collect all queries for which users click on the page in the results from the search logs. We then combine the queries into a new field for the Web page. We call this new field the *QueryText*. Each unique query in the *QueryText* field is called a *QueryText line*. Every line is associated with a weight that represents the prior relevance of this query to the page. The weight is determined by the query impressions (occurrences of a query in the query log) and click-through rate (CTR) on the page for this query. To utilize the new field in a search engine, we derive two sets of features from it: BM25F features and *n*-gram features. The BM25F features [18] are calculated over the entire document. In addition, *n*-gram features are derived from

Yahoo! Labs Technical Report No. YL-2009-007

this field by segmenting the QueryText lines into bigrams and trigrams. The n -gram features aim to provide better proximity measurement for long queries while leveraging the new field. We evaluate the two sets of features from the QueryText field in an existing commercial search engine with a real large-scale Web corpus. Our evaluation results show that both BM25F features and n -gram features obtained from user queries can significantly improve the relevance of the current search engine.

The contributions of this work are three-fold. First, we are the first to take user click information into consideration when calculating textual match features based on QueryText. While previous work also treats user queries as a new text field [17], the method does not incorporate user behavior recorded in query logs into the scoring scheme of the new field. We adopt a simple scheme of weighting using query impressions and CTR on documents in this work. We believe that our scoring scheme for QueryText opens avenues to further explore how to effectively integrate user click patterns with text weighting. Second, we evaluate BM25F features and n -grams features derived from the QueryText. While the BM25F features take into account the fielded structure of the documents, it does not handle proximity very well. For this reason, we examine various n -gram features designed to better address proximity issues for long queries. Finally, we conduct experimental evaluations in a commercial search engine environment. The data that we use comes from a full Web search log. We evaluate our approach by adding the proposed features to the search engine. To our knowledge, this is the first evaluation of the effectiveness of leveraging user queries for text matching within a real, large-scale production setting. Previous work simplifies the evaluations by using basic IR functions and simple document representation without other meta-data fields. Since such meta-data can be extremely useful for Web search, ignoring these fields can lead to inaccurate evaluations.

This paper is organized as follows. In Section 2, we review the related work and discuss the difference between our work and the existing literature. We then detail how the QueryText field is constructed in Section 3 and propose a set of textual features that can be extracted from the field in Section 4. In Section 5, we report our evaluation set up and experimental results. Finally, Section 6 concludes the paper and describes areas of future work.

2. Related Work

Recently, user query logs have garnered a great deal of interest in the research community. Research has shown these logs are a valuable source for improving relevance of search engines. Query log mining aims to discover useful signals for document ranking and query reformulations by mining the usage (mainly through clicks) patterns of users. Poblete and Baeza-Yates [13] classified web mining into three categories: content (text and multimedia of pages), structure (e.g. link structure) and usage. Query log mining belongs to usage mining.

Previous work on query log mining focuses on user click behavior. Agichtein *et al.* [1] and Joachims *et al.* [9] treat user click behavior as implicit feedback from users for search results, and use this feedback to influence document ranking in search engines. Specifically, Agichtein *et al.* extracted click information obtained from query logs to derive a number of features. The click-based features were then incorporated into a machine learned Web search ranking function. Our work is similar, in that we also integrate new features based on click information into a machine

Yahoo! Labs Technical Report No. YL-2009-007

learned ranking function. Different from their method, however, is that we use text matching features derived from queries and their associated clicks, instead of using the click information directly. Joachims *et al.* analyzed user clicks to derive reliable training data out of clicked documents to learn a ranking function. Similarly, Dupret and Piwowarski [7] proposed a user browsing model to estimate the probability of a document being examined by the user given the rank of the document and its distance from the last clicked document. The proposed approach provides a better model of user behavior for inferring document relevance from user clicks. All of the work mentioned here utilizes click information but does not consider the text of the queries associated with the clicks, which we address here.

Some previous work has tried to bridge the lexical gap between user queries and documents. Cui *et al.* [6] explicitly utilized text in queries and proposed using query logs as a source of query expansion. The approach connects queries and their clicked documents through query logs and extracted expansion terms from clicked documents for new queries. Billerbeck *et al.* [3] evaluated selecting query expansion terms from both past queries and anchor text. They showed that using past queries can achieve better retrieval effectiveness than using anchor text. However, they did not fully leverage query logs by considering user clicks. Poblete and Baeza-Yates [13] employed frequent-set mining technique to obtain query set (representative topical terms) from click-associated queries. Query sets are used as part of the document representation. However, they did not show how the query set affects search result ranking. Radlinski *et al.* describe another way to utilize user queries to minimize the lexical gap between queries and documents [14]. To combat matching short queries against online ads, an offline processing approach was used to find good query substitutes from query logs and similar ads. Finally, Craswell and Szummer [5] used a random walk model over the click graph to improve image retrieval.

There has also been work that investigates how to rank structured documents [12, 15]. Our work is related to this work, because we augment Web pages with a new textual field that consists of user queries. However, rather than developing new methods for ranking structured documents, we simply make use of standard similarity measures that have been proven to be effective in the past. In particular, we make use of BM25F, which was proposed by Robertson *et al.* [15]. The BM25F model effectively combines term frequencies from different document fields by non-linearly combining weighted term frequencies. Further details of our specific BM25F implementation are provided in Section 5.

The approach proposed by Xue *et al.* [17] is similar to our work. Their approach collects user queries for use as a document meta-data field. Documents are ranked by computing the similarity between incoming queries and the meta-data field. Our work differs from their work in the following aspects: (1) They did not evaluate the method with a commercial search engine, but instead used a simplified IR system. We argue that in a real-world search engine, a web page is reinforced by other meta-data, such as anchor text, and a large number of other features. The usefulness of user queries as additional meta-data could be diminished in the presence of these additional sources of information. Our evaluations are conducted using a commercial search engine with all other meta-data and features available. For this reason, we believe that our evaluation is more accurate in the context of large scale Web search. (2) Their work focused on utilizing similarity of documents and

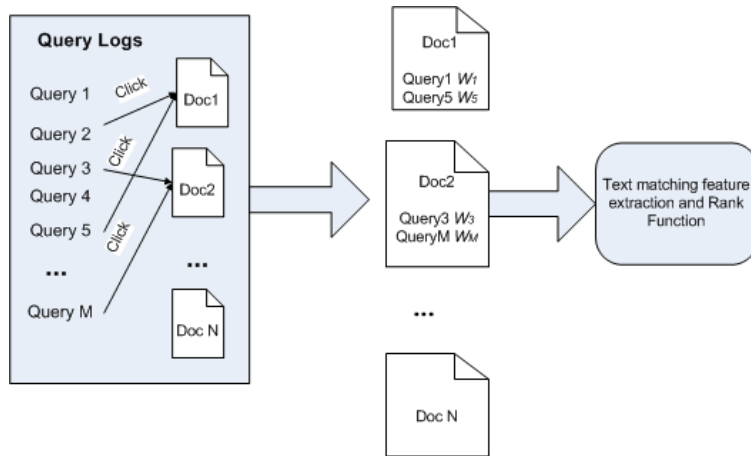


Figure 1: QueryText system architecture.

queries to combat the sparseness of document-query relations in query logs. Instead, we propose weighing user queries by user clicks and applying text matching against the weighted queries. We evaluate our methods using features derived from whole queries and n -grams. In addition, we argue that given the huge amount of query logs today, sparseness may not be a severe problem, since we observe good click coverage on our randomly sampled test queries.

3. Building Enriched Documents

In this section, we provide an overview of the QueryText system architecture and detail how the field is constructed from query logs.

3.1. Overview of Method

Figure 1 illustrates how we process the query logs and associate each query with clicked documents. Query logs record each user query session. In each session, a user issues a query and the search engine returns a list of Web pages. The user may click on some of the result pages or click nothing.

During query log processing, we first collect all queries that result in user's clicking one or more search results. We conduct a simple query normalization process which removes punctuation and extraneous whitespace. We also apply a small stopword list of 25 common words to the queries. To minimize the impact of noisy and random clicks, we filter the queries by thresholding query impressions (i.e., number of times a given query was issued over some period of time) and number of document clicks. Many malformed or misspelled queries which occur only once in the query logs are filtered out during this process. For some queries, users may accidentally click on some irrelevant pages, and thus we use the number of times a page was clicked for the query to filter out sporadic clicks. In this work, we keep queries with more than 5 impressions in a 6 month period.

Yahoo! Labs Technical Report No. YL-2009-007

Furthermore, the documents associated with these queries have to have received at least 2 clicks across all query sessions to be considered.

After collecting and filtering queries that have clicks, we collect all of the queries that led to clicks on a given document D . All such queries make up a new field, which we call QueryText, for D . This field augments the original document representation, which includes other fields such as title, body and anchor text. Each query in the field occupies one QueryText line and is associated with a weight that indicates the relevance of the query to the document. The weight, which we will now describe in detail, is calculated based on user clicks patterns.

3.2. QueryText Weighting using Clicks

User clicks on search results are a (potentially noisy) form of implicit feedback. In previous work, for instance, Agichtein *et al.* [1] and Joachims *et al.* [9] have used user behavior reflected in query logs to improve search result ranking. While their work exploits user behavior information directly for modeling relevance, we use user behavior signals to weigh QueryText lines. Specifically, we weigh each query in the QueryText using the following equations:

$$W(QT, D) = \log(I(QT)) \cdot CTR(QT, D)$$

$$CTR(QT, D) = \frac{Clicks(QT, D)}{I(QT)}$$

Given the QueryText line QT and document D , the weight of QT in the QueryText field of D is calculated as the product of the (log) number of impressions of QT and the click-through-rate (CTR) of QT on D . Here, CTR is defined as the number of clicks on D for query QT normalized by the number of impressions of QT . The rationale for employing such an equation is that we use query impressions to represent the popularity of a query and the CTR of a query on a document measures the document-specific importance of the query. If a QueryText line is a popular query and associated with many clicks on a document, that QueryText line is considered important for determining the relevance of the document.

This weighting scheme is simple to implement and fast to compute offline. It is certainly possible to explore more complex schemes for weighing QueryText lines based on a more careful examination of user click behavior, for instance, taking into account the fact that the user skipped some results before making a click [9]. However, we did not do so because this scoring scheme gives us a simple, yet robust, mechanism that can easily be incorporated into most existing search engines.

3.3. QueryText Examples

To illustrate the benefit of adding QueryText as a new field, we list some examples of anchor text and QueryText in Table 1 for several URLs. The examples show that QueryText is often complementary to anchor text in terms of annotating a Web page. For instance, the second URL has keywords such as “resume”, “common” and “mistakes” in QueryText, which expands the lexical

Yahoo! Labs Technical Report No. YL-2009-007

| baking.about.com/od/icecream/r/choco.htm | |
|---|--|
| chocolate ice cream 19.0 | homemade chocolate ice cream recipe 11.5 |
| chocolate 11.0 | homemade chocolate chip ice cream recipe 3.6 |

| career-advice.monster.com/.../Pitch-Yourself-for-an-Internship/ | |
|--|--|
| | cover letter for internship 3.36 |
| Sample Internship Cover Letter 1.0 | sample internship resume and cover letter 1.56 |
| | common cover letter mistakes 0.60 |

| journeytoforever.org/biodiesel_mike.html | |
|---|---------------------------------------|
| mike pelly's biodiesel method 7.33 | biodiesel recipe 3.68 |
| mike pelly's recipe 6.75 | biodiesel soap 2.05 |
| mike pelly's biodiesel recipe 4.25 | how to make your own diesel fuel 1.98 |

Table 1: Example anchor text (left column) and QueryText (right column) for several URLs. The weight associated with each line is also shown.

coverage of the document. Sometimes, QueryText gives different emphasis on keywords than anchor text. If we look at the third URL in Table 1, we can see that anchor text biases on “Mike Pelly” while QueryText emphasizes more on “biodiesel”, which is more aligned with user intent. Since QueryText comes from user queries, it bridges the gap between the vocabulary of users and document keywords well. The empirical effectiveness of incorporating QueryText in Web search will be demonstrated in Section 5.

4. QueryText Features

Text matching features measure how well a query matches against the textual representation of a document. While current commercial search engines employ many other important features, including spam and link analysis features, text matching features are the most important features in their ranking functions. Most web ranking functions perform text match against the various document fields, with the contribution of each field weighted according to its general importance. It has been shown that anchor text and title fields play a much more important role than the body field for many Web search queries [12].

We propose two sets of text matching features that can be derived from the QueryText field for each document – BM25F features and query n -gram features. The BM25F feature measures how well an incoming query matches the entire document, including the QueryText lines. We also introduce a set of query n -gram features. We segment the QueryText lines into bigrams and trigrams and derive a number of aggregated features based on query n -grams. These n -gram features help improve matching because they act as a simple form of proximity matching, which is not captured in the standard BM25F model.

Our features are designed to support full query coverage. That is, they can be applied to any incoming query, not just queries that we have seen before. A large number of so-called tail queries

Yahoo! Labs Technical Report No. YL-2009-007

consist of 3 or more query words. Long queries tend to yield fewer (poor recall), and lower quality (poor precision), results than short queries. As we will show, matching such queries against the QueryText section using BM25F and n -gram features result in strong retrieval effectiveness.

4.1. BM25F Features

Our BM25F feature implementation follows the description provided in [18]. A structured document D (a Web page in this work) can be represented as a series of fields:

$$D = (d_1, d_2, \dots, d_K)$$

where d_1 may represent the title of document D , d_2 may be the body of D , and so on. The QueryText field proposed here is simply treated as another field within the document, except that the lines within the field have weights associated with them.

The score between an incoming query Q and a structured document D is then computed as:

$$BM25F(Q, D) = \sum_{t \in Q} \frac{wt(t, D)}{k_1 + wt(t, D)} \cdot \log \frac{N - df_t + 0.5}{N + 0.5}$$

where N represents the number of documents in the corpus, df_t is the number of documents that contain term t , and $wt(t, D)$ denotes the weight of query term t in document D . Here, $wt(t, D)$ is computed as a weighted sum of term weights from each field as follows:

$$wt(t, D) = \sum_{i=1}^K wt(d_i) \cdot \hat{wt}(t, d_i)$$

where $wt(d_i)$ is the weight of field d_i and $\hat{wt}(t, d_i)$ is estimated as:

$$\hat{wt}(t, d_i) = \frac{wt(t, d_i)}{1 + b_i \left(\frac{l(d_i)}{avgl(d_i)} - 1 \right)}$$

where $wt(t, d_i)$ is the term weight of t in field d_i and l is the length of the field. For fields other than QueryText, $wt(t, d_i)$ is simply the term frequency of the term in the field. For QueryText, since each line is weighted, we use the following equation to calculate $\hat{wt}(t, d_{QT})$:

$$\hat{wt}(t, d_{QT}) = \sum_{L \in QT} wt(L) \cdot tf(t, L) \cdot \alpha^{x(Q, L)} \cdot \beta^{m(Q, L)}$$

Here, L stands for a line from the QueryText field and $wt(L)$ is the weight of the line calculated as described previously (i.e., $W(QT, D)$ in Section 3.2). Furthermore, $tf(t, L)$ denotes the term frequency of term t in the line L . Both α and β are pre-tuned constants that penalize extra and missing query terms in the line L and $m(L, Q)$ and $x(L, Q)$ represent the number of extra and missing query terms, respectively. This weighting scheme was shown to be effective in [2].

Yahoo! Labs Technical Report No. YL-2009-007

| |
|---|
| Max. weight of matched bigrams |
| Max. weight of matched trigrams |
| Min. weight of matched bigrams |
| Min. weight of matched trigrams |
| Avg. weight of matched bigrams |
| Avg. weight of matched trigrams |
| Proportion of matched bigrams in the query |
| Proportion of matched trigrams in the query |

Table 2: Summary of n -gram features.

4.2. Query n -gram Features

In addition to the BM25F feature just described, we also introduce a set of query n -gram features. The n -gram features are directly derived from the QueryText lines and they inherit the same weights (described in Section 3.2) of the lines from which they originate. For simplicity, we only consider query bigrams and trigrams. For instance, the query “northern California car sale” generates bigrams “northern California”, “California car” and “car sale”, as well as trigrams “northern California car” and “California car sale”. The weights of each of these bigrams and trigrams for a given document are equal to the weight of the QueryText line “northern California car sale” in that document.

In this setting, the QueryText conceptually consists of queries, together with a list of n -grams, each with an assigned weight. Given an incoming query, the query is segmented into bigrams and trigrams which are then matched against the n -grams in the QueryText field to compute the feature values. Table 2 lists all of the n -gram features that we consider here. These features attempt to capture various aspects of how well the query n -grams match the QueryText n -grams. The features derived from the matched bigrams and trigrams are used as input to the ranking function, as will be described shortly.

5. Empirical Evaluation

In this section, we describe our empirical evaluation of the proposed approach. We first describe the evaluation metrics and setup, and then detail the evaluation results.

5.1. System and Data

To evaluate the effectiveness of our proposed text matching features, we implement our methods on top of a commercial search engine that builds its ranking function around a machine learning framework [19]. We evaluate the effectiveness of BM25F features and n -gram features for QueryText with a large-scale Web training and test corpus. To our knowledge, this is the first time that similar ideas are evaluated with Web-scale data.

Yahoo! Labs Technical Report No. YL-2009-007

| Query Length (in words) | 1 | 2 | 3 | 4+ |
|-------------------------|-------|-------|-------|-------|
| Number of Queries | 4,107 | 7,486 | 5,394 | 6,059 |

Table 3: Query length distribution.

Our data set consists of 23,046 English queries randomly sampled from query logs. The number of queries for each word length is listed in Table 3. For each query, we have an average of 22.75 URLs, resulting in 524,418 query-URL pairs. Each URL has been judged by human editors and given a rating of Perfect, Excellent, Good, Fair, or Bad to indicate how relevant it is to the corresponding query. This data is randomly split in two to produce training and test sets respectively.

We adopt commonly used metrics, namely DCG (Discounted Cumulated Gain) and NDCG (Normalized DCG) [8], to measure retrieval effectiveness. The DCG at rank K is computed as:

$$DCG@K(Q) = \sum_{i=1}^K \frac{g(i)}{\log(1+i)}$$

where $g(i)$ denotes the grade of judgment for the result at rank i and K is the maximum depth of results to consider. In this work, we consider both DCG-1 and DCG-5, which are commonly used to evaluate Web search engines. Furthermore, NDCG is defined as:

$$NDCG(Q) = \frac{DCG(Q)}{IDCG(Q)}$$

$$DCG(Q) = \sum_{i=1}^{N(Q)} \frac{g(i)}{\log(1+i)}$$

where $N(Q)$ stands for the number of results ranked for query Q and $IDCG(Q)$ is the “ideal DCG”, which is the DCG obtained if the results for Q were ranked perfectly (i.e., in decreasing order of their grade).

To generate the QueryText fields, we accumulated 6 months of recent query logs. We overlapped the URLs in our data set with the query logs to obtain 60,391 unique URLs that have clicks. The 60,391 URLs covers 12.54% of 481,616 unique URLs in our data set. On average, 100 queries are associated with each of the URLs.

5.2. Evaluation Results

To provide both academic and industrial benchmarks, we conduct two separate evaluations. The first evaluation looks at a BM25F only ranking function, which is commonly used in the academic community for Web search evaluations, while the second evaluates our proposed features when used within a real commercial search engine with many other features.

Yahoo! Labs Technical Report No. YL-2009-007

| Model | DCG-1 | DCG-5 | NDCG |
|-------------------|------------------------|------------------------|------------------------|
| BM25F | 3.2649 | 7.5757 | 0.8255 |
| BM25F + QueryText | 3.3018 (+1.30%) | 7.6383 (+0.83%) | 0.8275 (+0.24%) |

Table 4: Comparison of DCG and NDCG Between Baseline and Baseline with BM25F Features for QueryText (significant improvement in bold, $p < 0.01$ for t-test).

5.2.1. BM25F Ranking Function Our first evaluation, as just described, employs a BM25F-only ranking function. This can be considered a rough approximation to an actual search engine that only does not have any non-textual features, such as those derived from the link graph features and spam features. The reason to include this evaluation is to demonstrate the direct benefit of adding QueryText as additional matching field for document retrieval. Similar ranking functions have been used as baseline in previous work [1]. To be consistent with other fields that do not have n -gram features, we exclude n -gram features from this evaluation.

In this experiment, the baseline is BM25F computed over all of the standard Web document fields, such as title, body, and anchor text. The BM25F+QueryText setting augments each document with the QueryText field.

It is important to note that all of the BM25F parameters, both in the baseline and QueryText version of the model, are optimized to maximize NDCG on the training set. We iteratively search for the optimal values of k_1 and b_f , each at a time. In each iteration, we fix the value of one parameter to be the one found in the last iteration while optimizing the others. We repeat the iterations until the NDCG value stops increasing, and the values have converged. This is similar to the BM25F parameter optimization approach described in [18].

The results of our experiments are shown in Table 4. The results show that the BM25F features on QueryText significantly improve the search relevance according to all metrics. It is worth noting that the relative improvement may seem small compared to other studies, but the improvement is quite large for this particular data set. The reason is that we use a very large data set with over 10,000 queries in the test set (we randomly split the whole data set into training and test sets). Such a sample includes many head queries, which can be considered solved, with little room for improvement, and many tail queries, which can be considered hard, and therefore difficult to improve greatly.

Our experiments show that around 40% of the test queries are improved as the result of the QueryText field, which shows that such features improve a large swath of queries. All of the bolded values in the table are significantly better than the baseline with p -value less than 0.01 according to a paired one-tailed t -test. We see clearly that adding the new text matching features boosts the baseline by 1.30% in DCG-1, 0.83% in DCG-5 and 0.24% in NDCG over all queries.

We also examine the effect of the QueryText section on query length. The results in Table 5 show DCG-1, DCG-5, and NDCG stratified by query length. By breaking the queries down by length, we find that the QueryText BM25F features improve the baseline the most in query length of 2 and 3, and the least for 4+ word queries. For 4+ word queries, the new features significantly improve DCG-1 and DCG-5, but only moderately impact NDCG. We attribute this finding to the

Yahoo! Labs Technical Report No. YL-2009-007

| Query Length | Model | DCG-1 | DCG-5 | NDCG |
|--------------|-------------------|------------------------|------------------------|------------------------|
| 1 word | BM25F | 3.9623 | 9.2638 | 0.8299 |
| | BM25F + QueryText | 3.9828 (+0.52%) | 9.3248 (+0.66%) | 0.8321 (+0.26%) |
| 2 word | BM25F | 3.7343 | 8.6913 | 0.8293 |
| | BM25F + QueryText | 3.7748 (+1.09%) | 8.7694 (+0.90%) | 0.8318 (+0.29%) |
| 3 word | BM25F | 3.0938 | 7.1991 | 0.8226 |
| | BM25F + QueryText | 3.1388 (+1.46%) | 7.2733 (+1.03%) | 0.8253 (+0.33%) |
| 4+ word | BM25F | 2.3661 | 5.3920 | 0.8207 |
| | BM25F + QueryText | 2.4023 (+1.53%) | 5.4264 (+0.64%) | 0.8214 (+0.09%) |

Table 5: Impact of Query Lengths (4+ word means 4-word and longer queries; significant improvement in bold, $p < 0.01$ for t-test).

fact that coverage of QueryText for shorter queries is larger than longer queries. According to our statistics, the unique queries that match at least one query term in the QueryText field constitute 68.83%, 68.12%, 61.68% and 52.08%, respectively for query lengths of 1, 2, 3, and 4+. It is obvious that the coverage of long queries is much lower than short queries. Furthermore, the BM25F model does not account for term proximity, which may hurt longer queries. We expect that the query n -gram features, when used in conjunction with BM25F, can lead to further improvements in retrieval effectiveness. Indeed, our next set of experiments test this hypothesis.

5.2.2. Machine Learned Ranking Function Our second evaluation integrates both our BM25F feature and n -gram features into an existing commercial search engine to demonstrate their utility in a real web search environment. The ranking function of the search engine is trained on human judged query-url pairs, as described previously, and employs gradient boosted decision trees as learning method, as proposed in [19]. This ranking function uses hundreds of features, including query-dependent, query-independent, click-related features, etc. QueryText BM25F features and n -gram features are treated as additional features for the ranking function. In this way, we can demonstrate the effectiveness of QueryText-based features in the presence of other features, such that the interaction with other features can be understood. For example, some documents with partial matches of n -gram QueryText features may not be relevant due to other reasons, such as being a spam page. Having all features plus QueryText features together trained by the ranking function will address such problem and provide a more realistic evaluation.

The baseline model here is the learned ranking function without QueryText features, while the Baseline+QueryText model uses the QueryText-based features. The training and test splits in both evaluations are the same, as are all of the tunable model parameters, thereby making the evaluation as fair as possible.

Table 6 shows the results of our evaluation. Similar to the BM25F-only evaluation, we observe a statistically significant improvement over all metrics when we add QueryText features to the machine learned ranking function. In this experiment, as describe before, we use a full-fledged commercial search engine with a full set of features. In this setup, we are more confident in gauging the effectiveness of the QueryText-based features in a real-world environment, including their

Yahoo! Labs Technical Report No. YL-2009-007

| Configuration | DCG-1 | DCG-5 | NDCG |
|----------------------|------------------------|-------------------------|------------------------|
| Baseline | 4.5660 | 10.7504 | 0.6152 |
| Baseline + QueryText | 4.6322 (+1.67%) | 10.9500 (+1.86%) | 0.6294 (+2.31%) |

Table 6: Comparison of DCG and NDCG between Baseline and Baseline with QueryText Features (significant improvement in bold, $p < 0.01$ for t-test).

| Query Length | Model | DCG-1 | DCG-5 | NDCG |
|--------------|----------------------|------------------------|-------------------------|------------------------|
| 1 word | Baseline | 6.6509 | 15.2223 | 0.7383 |
| | Baseline + QueryText | 6.6811 (+0.45%) | 15.3895 (+1.03%) | 0.7460 (+1.05%) |
| 2 word | Baseline | 5.3319 | 12.6531 | 0.6734 |
| | Baseline + QueryText | 5.3987 (+1.07%) | 12.8193 (+1.23%) | 0.6835 (+1.50%) |
| 3 word | Baseline | 4.2214 | 10.0383 | 0.5944 |
| | Baseline + QueryText | 4.2912 (+1.65%) | 10.2533 (+2.24%) | 0.6104 (+2.68%) |
| 4+ word | Baseline | 2.7837 | 6.7288 | 0.4989 |
| | Baseline + QueryText | 2.9123 (+4.62%) | 6.9784 (+3.72%) | 0.5198 (+4.21%) |

Table 7: Impact of Query Lengths (4+ word means 4-word and longer queries; significant improvement in bold, $p < 0.01$ for t-test).

possible interaction with other features. For example, we suspected that our QueryText features may already be mostly explained by other click-related and anchor text features, which would diminish their usefulness. However, as we can see from the results, this is not the case. Indeed, our positive results suggest that the new QueryText features bring in new, useful signals for relevance. Combining BM25F and n -gram features from QueryText with other features in the machine learned model improves all queries by 1.86% in DCG-5 and 2.31% in NDCG, both of which are statistically significant (p -value < 0.01).

As with the BM25F-only evaluation, we also break down retrieval effectiveness by query length. These results are shown in Table 7. By adding n -gram features to BM25F features, we can see an interesting trend that n -gram features improve more on long queries, while the BM25F-only ranking function alone improves more on shorter queries. This supports our hypothesis that n -gram features, which implement a simple form of term proximity matching, can be used to improve longer queries, where proximity plays a more critical role. Indeed, we observe large improvements over 3 and 4+ word queries, which are improved by 2.24% and 3.72% in DCG-5, respectively. The new features also improve NDCG for 3 and 4+ word queries by 2.68% and 4.21%, all of which are statistically significant.

We also compared the effectiveness of trigram features with bigram features. We observe that trigram features obtain better improvement than bigram features on long queries (4+ word queries). We attribute it to the fact that trigrams have less ambiguity than bigrams and thus provide more accurate match against user queries. However, if we further experiment with 4-grams and 5-grams, the performance starts to drop dramatically due to the low coverage of longer n -grams on new queries. As such, we would argue that trigram features are robust for Web search features, especially when applied to longer queries. This was also tested and shown to hold by Bai *et al.* [2].

5.3. Deeper Analysis

In this section, we undertake a deeper analysis of our results to develop a better understanding of the usefulness of QueryText. To be fair, we highlight several examples of where QueryText successfully improves search relevance, and also provide examples where it degrades relevance. These case studies were undertaken by manually analyzing our results to understand *why* certain queries observed improved or degraded relevance. We now summarize our findings.

Our analysis revealed that relevant results for some queries have little, or no, anchor text lines. Often, such relevant pages reside in blogs and online discussion forums. As such, these pages are not popularly linked from external web sites, despite the fact that they are excellent results for some queries which may not be answered by more popular pages. For example, consider the query “tanker companies”. One of the good results is <http://oiltankers.blogspot.com/> which is highly relevant in general. This is a blog page and only has two anchor text lines, which are its URL and “Oil Tankers”. Without QueryText, this page is ranked out of the top 20 results. In contrast, some news articles which have many query term occurrences in the pages and the anchor text are ranked much higher than this page. However, the news pages are not relevant as they mainly describe specific events related to tankers. The good page from above has over 50 lines in QueryText, each of which provides additional and accurate annotations of the page. Example QueryText lines for the page include “crude oil tankers”, “oil tanker companies” and “world’s largest oil tanker”. In this way, the good result page is ranked as the top result with QueryText. Therefore, QueryText is useful for overcoming the anchor text sparseness problem described in [11].

Additionally, some queries are generic and ambiguous. The relevant pages for such queries may not have enough lexical overlap with the user queries which may cause them to not appear in the result list. One example of such a query is “song name finder”. A highly relevant page for this query is <http://findmeatune.com/>. However, neither the page content nor the anchor text of the page has phrases like “find song name” or “song name finder”. The page, however, has QueryText lines such as “find song name by lyrics” and “song name finder by lyrics” which are weighted highly by our proposed method. As such, the relevant page is ranked as the top result when taking QueryText into account.

Despite QueryText’s success in the difficult queries illustrated above, it may degrade the relevance of search results for some queries. Some of such queries are navigational queries. For instance, for the query “quicken tax”, one of the highly relevant pages, <http://www.turbotax.com/>, is ranked out of the top 5 results due to the fact that other less relevant pages containing both “quicken” and “tax” in the QueryText are highly ranked. One solution to this deficiency is that to train different ranking functions that exclude (or lower the weight of) QueryText for different categories of queries, such as navigational, where we expect the anchor text alone to be sufficient for ranking.

6. Conclusions and Future Work

In this paper, we proposed a new framework for representing Web pages. We enrich Web page representations with a new field of aggregated user queries, weighted according to search result page clicks. We show that various text matching features, such as BM25F and n -gram features,

Yahoo! Labs Technical Report No. YL-2009-007

can easily be extracted from the new field. We evaluated our approach with an existing commercial search engine. The results of two detailed evaluations show that our approach significantly and consistently improves the relevance of the current search engine on real Web data, thereby proving the usefulness of QueryText.

User queries and clicks have long been studied to improve relevance of Web search. However, previous work focused more on using user clicks as implicit feedback and employing user queries to identify query reformulations. Our work differs from previous work in that we directly aggregate user queries into a new document field and extract text matching features from the field. More importantly, we take user click behavior into account in weighing the lines of the new field. The approach effectively leverages user queries as a source of page annotation, as well as clicks associated with the queries on the Web pages. To our knowledge, we are the first to evaluate such method using a real search engine with Web-scale data corpus.

There are several potential directions to extend this work. First, it may be possible to employ more complex text matching techniques to derive more effective features. In this work, for simplicity, we adopt simple text match features, such as BM25F scoring. However, as we noted, BM25F alone does not capture proximity. BM25F features only address the occurrences of query terms, while not considering order and adjacency of query terms in the field. We partially address this problem by introducing n -gram features. It would be worthwhile to explore more proximity measures [2, 10, 16] in deriving new features out of the field of QueryText. Second, it would be interesting to explore different types of user data. In this study, we utilize user search logs as the sole source for discovering useful queries to annotate Web pages. Other user data sources, such as toolbar logs, can also be employed to find such queries with clicks. Finally, it may be worth exploring more signals in weighing queries for text matching. While we adopt relatively simple approach in weighing queries by user clicks, it is worthwhile to explore more signals embedded in query logs to weigh queries on their relevance to the clicked Web pages. Joachims *et al.* [9] proposed complex user behavior model that takes into account skipping some results in click patterns to accurately model relevance based on clicks. Other signals, such as the duration of clicks on specific Web pages is also deemed valuable in determining if a clicked Web page is relevant to the query.

References

- [1] E. Agichtein, E. Brill, and S. Dumais. Improving web search ranking by incorporating user behavior information. In *Proc. 29th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 19–26, New York, NY, USA, 2006. ACM.
- [2] J. Bai, Y. Chang, H. Cui, Z. Zheng, G. Sun, and X. Li. Investigation of partial query proximity in web search. In *Proc. 17th Intl. Conf. on World Wide Web*, pages 1183–1184, New York, NY, USA, 2008. ACM.
- [3] B. Billerbeck, F. Scholer, H. E. Williams, and J. Zobel. Query expansion using associated queries. In *Proc. 12th Intl. Conf. on Information and Knowledge Management*, pages 2–9, New York, NY, USA, 2003. ACM.

Yahoo! Labs Technical Report No. YL-2009-007

- [4] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.
- [5] N. Craswell and M. Szummer. Random walks on the click graph. In *Proc. 30th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 239–246, New York, NY, USA, 2007. ACM.
- [6] H. Cui, J.-R. Wen, J.-Y. Nie, and W.-Y. Ma. Probabilistic query expansion using query logs. In *Proc. 11th Intl. Conf. on World Wide Web*, pages 325–332, New York, NY, USA, 2002. ACM.
- [7] G. E. Dupret and B. Piwowarski. A user browsing model to predict search engine click data from past observations. In *Proc. 31st Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 331–338, New York, NY, USA, 2008. ACM.
- [8] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002.
- [9] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting click-through data as implicit feedback. In *Proc. 28th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 154–161, New York, NY, USA, 2005. ACM.
- [10] D. Metzler and W. B. Croft. A Markov random field model for term dependencies. In *Proc. 28th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 472–479, 2005.
- [11] D. Metzler, J. Novak, H. Cui, and S. Reddy. Building enriched document representations using aggregated anchor text. In *Proc. 32nd Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, page To appear, 2009.
- [12] P. Ogilvie and J. Callan. Combining document representations for known-item search. In *Proc. 26th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 143–150, 2003.
- [13] B. Pobleto and R. Baeza-Yates. Query-sets: using implicit feedback and query patterns to organize web documents. In *Proc. 17th Intl. Conf. on World Wide Web*, pages 41–50, New York, NY, USA, 2008. ACM.
- [14] F. Radlinski, A. Broder, P. Ciccolo, E. Gabrilovich, V. Josifovski, and L. Riedel. Optimizing relevance and revenue in ad search: a query substitution approach. In *Proc. 31st Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 403–410, New York, NY, USA, 2008. ACM.
- [15] S. Robertson, H. Zaragoza, and M. Taylor. Simple bm25 extension to multiple weighted fields. In *Proc. 13th Intl. Conf. on Information and Knowledge Management*, pages 42–49, 2004.

Yahoo! Labs Technical Report No. YL-2009-007

- [16] T. Tao and C. Zhai. An exploration of proximity measures in information retrieval. In *Proc. 30th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 295–302, New York, NY, USA, 2007. ACM.
- [17] G.-R. Xue, H.-J. Zeng, Z. Chen, Y. Yu, W.-Y. Ma, W. Xi, and W. Fan. Optimizing web search using web click-through data. In *Proc. 13th Intl. Conf. on Information and Knowledge Management*, pages 118–126, New York, NY, USA, 2004. ACM.
- [18] H. Zaragoza, N. Craswell, M. Taylor, S. Saria, and S. Robertson. Microsoft Cambridge at TREC 13: Web and hard tracks. In *Proc. 13th Text REtrieval Conference*, 2004.
- [19] Z. Zheng, K. Chen, G. Sun, and H. Zha. A regression framework for learning ranking functions using relative relevance judgments. In *Proc. 30th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 287–294, New York, NY, USA, 2007. ACM.