Search Advertising using Web Relevance Feedback

Andrei Z. Broder†, Peter Ciccolo†, Marcus Fontoura‡,
Evgeniy Gabrilovich†, Vanja Josifovski†, Lance Riedel†

† Yahoo! Research, 2821 Mission College Blvd, Santa Clara, CA 95054, USA
‡ Computer Science Department, PUC-Rio, Rio de Janeiro, Brazil
† {broder | ciccolo | gabr | vanjaj | riedell}@yahoo-inc.com | ‡ mfontoura@inf.puc-rio.br

ABSTRACT

The business of Web search, a $10 billion industry, relies heavily on *sponsored search*, whereas a few carefully-selected paid advertisements are displayed alongside algorithmic search results. A key technical challenge in sponsored search is to select ads that are relevant for the user's query. Identifying relevant ads is challenging because queries are usually very short, and because users, consciously or not, choose terms intended to lead to optimal Web search results and not to optimal ads. Furthermore, the ads themselves are short and usually formulated to capture the reader's attention rather than to facilitate query matching.

Traditionally, matching of ads to queries employed standard information retrieval techniques using the bag of words approach. Here we propose to go beyond the bag of words, and augment both queries and ads with additional knowledge-rich features. We use Web search results initially returned for the query to create a pool of relevant documents. Classifying these documents with respect to an external taxonomy and identifying salient named entities give rise to two new feature types. Empirical evaluation based on over 9,000 query-ad pairwise judgments confirms that using augmented queries produces highly relevant ads. Our methodology also relaxes the requirement for each ad to explicitly specify the exhaustive list of queries ("bid phrases") that can trigger it.

Categories and Subject Descriptors

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

General Terms: Algorithms, economics, performance

Keywords: Online advertising, relevance, Web search

1. INTRODUCTION

The Web has become an integral part of our lives: people around the world use it for information, entertainment, shopping, communication, and many other activities. Navigating the Web without search engines would be impossible and more than 2 billion searches are performed every day [9].

The prevailing business model of Web search relies heavily on *sponsored search*, whereas a few carefully-selected paid advertisements are displayed alongside algorithmic (or *organic*) search results. There is a fine but important line between placing ads reflecting the query intent, and placing unrelated ads: users may find the former beneficial, as an additional source of information or an additional Web navigation facility, while the latter are likely to annoy the searchers and hurt the user experience (we discuss Web advertising in more detail in Section 2).

Identifying relevant ads is far from trivial, mainly because search queries are so short (the average query is only about 2.5 words long), and because users, consciously or not, choose query terms intended to retrieve the best search results rather than the best ads.

In the realm of Web search (and more generally within the field of information retrieval), there have been a number of studies on query augmentation [1, 3, 31, 48, 50], but as far as we know, no studies focused on query expansion for ad search. The latter task is notably more difficult than general query expansion since ads are typically quite short and are often formulated as abrupt, non-grammatical phrases intended to capture reader's attention rather than to facilitate query matching. Consequently, the usefulness of the ad corpus itself for query expansion is limited, hence we opted to explore ways of query augmentation using external sources of knowledge, including Web search results and a large taxonomy of commercial topics.

Given the original query (called the *Web query* in the sequel), we first send it to a Web search engine, and then use the returned top-scoring pages to gather additional knowledge about the query. We use this knowledge to create an augmented query (called the *ad query* in the sequel), which is evaluated against the ad corpus to retrieve relevant ads for the original Web query. Of course, short queries are also difficult for Web search; however, modern search engines use a huge amount of additional knowledge such as past query statistics, link analysis, page popularity, anchor text, and click-through data, and thus can return decent results even for very short inputs. Thus, the highest-scoring search results are often quite good, and so we use them for query augmentation within a blind relevance feedback approach.

Such additional knowledge becomes invaluable when processing malformed queries, such as misspelled ones or queries in which several words are glued together. For example, when the user types a misspelled query "car insuance", the

CIKM'08, October 26–30, 2008, Napa Valley, California, USA.
Copyright 2008 ACM 978-1-59593-991-3/08/10 ...$5.00.

search engine automatically corrects the spelling mistake and the search results for the corrected query can also be used for matching ads.

Historically, the mainstream approach to textual document retrieval has been based on the bag of words paradigm, where both the query and the documents to be retrieved are represented as vectors of word-based features [41], whose values are computed using a variant of the TFIDF weighting scheme [39]. In this work we go beyond the bag of words by using search results returned for the Web query to construct three classes of features that together form the ad query. For the first class of features, we pool together the words (unigrams) that occur within the result pages, and select the most representative ones to be used in addition to the original query words. The second class of features is based on our previous work on query classification using Web search results [7]. In that work, we classified the search results with respect to a large external taxonomy of over 6,000 nodes, and then used voting to determine the best classifications for the original query. Here we apply a similar technique to classify the Web query into relevant classes, which then define new features of the ad query. The third class of features is defined by a large lexicon of phrases, built by analyzing the set of all Web pages crawled by the underlying search engine. We identify all the entries of this lexicon that appear in the search results for the Web query, and then retain the most representative ones as additional features.

Ads undergo a similar processing, consisting of word analysis, taxonomy classification, and extraction of lexicon phrases. When both queries and ads are represented in this augmented space of features, their matching amounts to computing conventional similarity metrics such as cosine [53].

The contributions of this study are fourfold:

- We propose a methodology for cross-corpora query expansion for sponsored search, where we use one corpus (the Web) to augment queries to be evaluated against another corpus (the ads). While cross-corpora expansion techniques have been studied previously, to the best of our knowledge this is the first study to apply such methods to ad selection in sponsored search.

- We propose methods for defining a richer representation of both queries and ads by constructing new features based on external knowledge.

- In order to facilitate matching of relevant ads to queries under the bag of words approach, advertisers normally pre-define the queries ("bid phrases") for which it would be desirable to display a given ad. This approach, however, restricts the ad display to a relatively small set of queries. In this work we relax the requirement that advertisers explicitly specify "bid phrases"; instead, we use the entire contents of the ad to identify queries for which it should be shown.

- We present an evaluation of an end-to-end methodology for ad selection in sponsored search, based on query expansion coupled with an inverted ad index that can evaluate long queries.

Using the classification-based and phrase-based features facilitates thematic matching that goes beyond the simple bag of words approach and captures deeper semantic similarity. Our experimental evaluation confirms that using these additional features greatly improves the accuracy of ad matching, resulting in more relevant ads.

2. BACKGROUND: TEXTUAL ADVERTISING ON THE WEB

A large part of the Web advertising market consists of *textual ads*, the ubiquitous short text messages usually marked as "sponsored links". There are two main channels for distributing such ads. *Sponsored search* (or *paid search advertising*) places ads on the result pages of a Web search engine, where ads are selected to be relevant to the search query (see [16] for a brief history of the subject). All major Web search engines (Google, Microsoft, Yahoo!) support sponsored ads and act simultaneously as a Web search engine and an ad search engine. *Content match* (or *contextual advertising*) places ads on third-party Web pages. Today, almost all of the for-profit non-transactional Web sites[1] rely at least to some extent on contextual advertising revenue. Content match supports sites that range from individual bloggers and small niche communities to large publishers such as major newspapers.

In this paper we focus on sponsored search. However, we believe that additional knowledge-based features are also likely to be beneficial for content match, and plan to investigate this direction in future work.

Sponsored search is an interplay of three entities. The **advertiser** provides the supply of ads. Usually the activity of the advertisers is organized around *campaigns*, which are defined by a set of ads with a particular temporal and thematic goal (e.g., sale of digital cameras during the holiday season). As in traditional advertising, the goal of the advertisers can be broadly defined as promotion of products or services. The **search engine** provides "real estate" for placing ads (i.e., allocates space on search results pages), and selects ads that are relevant to the user's query. **Users** visit the Web pages and interact with the ads.

The prevalent pricing model for textual ads is that the advertisers pay for every click on the advertisement (pay-per-click or PPC). There are also other models, such as pay-per-impression, where the advertiser pays for the number of exposures of an ad, and pay-per-action, where the advertiser pays only if the ad leads to a sale or similar completed transaction. In this paper we deal with the PPC model, which is the one most often used in practice.

The amount paid by the advertiser for each sponsored search click is usually determined by an auction process [14]. The advertisers place *bids* on a search phrase, and their position in the column of ads displayed on the search results page is determined by their bid. Thus, each ad is annotated with one or more *bid phrases*. In addition to the bid phrase, an ad also contains a *title* usually displayed in bold font, and a *creative*, which is the few lines of text, usually shorter than 120 characters, displayed on the page. Naturally, each ad contains a URL to the advertised Web page, called the *landing page*.

In the model currently used by all the major search engines, bid phrases serve a dual purpose: they explicitly specify queries that the ad should be displayed for and simultaneously put a price tag on a click event. Obviously, these price tags could be different for different queries. For ex-

[1] Non-transactional sites are those that do not sell anything directly.

ample, a contractor advertising his services on the Internet might be willing to pay a small amount of money when his ads are clicked from general queries such as "home remodeling", but higher amounts if the ads are clicked from more focused queries such as "hardwood floors" or "laminate flooring". Most often, ads are shown for queries that are expressly listed among the bid phrases for the ad, thus resulting in an *exact match* (i.e., identity) between the query and the bid phrase. However, it might be difficult (or even impossible) for the advertiser to list all the relevant queries ahead of time. Therefore, search engines also have the ability to analyze queries and modify them slightly in an attempt to match pre-defined bid phrases. This approach, called *broad* (or *advanced*) match, facilitates more flexible ad matching, but is also more error-prone, and only some advertisers opt for it. Nonetheless, bid phrases remain a mandatory component of the ad definition.

Given a query q, the revenue from a click can be estimated as

$$R = \sum_{i=1..k} P(click|q, a_i) \cdot price(a_i, i),$$

where k is the number of ads displayed on the page with search results for q and $price(a_i, i)$ is the click price of the ad a_i at position i. The price in this model depends on the set of ads presented on the results page. Several models have been proposed to determine this price, most of them based on generalizations and variants of second price auctions (for more details, see [14] and references therein). For simplicity, in this paper we ignore the pricing model and concentrate on finding ads that will maximize the first term of the product, that is, we search for

$$\arg\max_i P(click|q, a_i).$$

Furthermore, we assume that the probability of a click for a given ad and query is determined by the ad's relevance score with respect to the query, thus ignoring the positional effect of the ad placement on the results page. We assume that this is an orthogonal factor to the relevance component, and could be easily incorporated in the model.

3. METHODOLOGY

In this section we present our methodology for using the Web for constructing new features for representing queries and ads. This approach allows us to leverage external knowledge available to search engines in order to create more informative features for matching ads to queries. Furthermore, by using features that characterize the entire ad rather than only its bid phrase, we relax the requirement for advertisers to explicitly specify bid phrases.

3.1 System overview

The input to our system is a search (or "Web") query, and the output is a set of ads that are relevant to this query. Processing the input query involves two main phases. In the first phase, we conduct a Web search with the original query, and analyze the top-scoring results obtained for it. We use these search results to augment the Web query and construct an ad query, which is then evaluated against an index of ads. Figure 1 presents a high-level view of the information flow.

We represent ad queries and ads in three distinct feature spaces that are formed using three different kinds of features, namely, unigrams, classes, and phrases. Thus, each object is represented as a feature vector, which is composed of three sub-vectors, each of which is normalized and scored separately. Let q be a query, then its feature vector is defined as follows: $v_q = \langle uq_1, \ldots, uq_{|U|}, cq_1, \ldots, cq_{|C|}, pq_1, \ldots, pq_{|P|} \rangle$, where U, C and P are the sets of unigrams, classes and phrase features, respectively. Given an ad a and its vector $v_a = \langle ua_1, \ldots, ua_{|U|}, ca_1, \ldots, ca_{|C|}, pa_1, \ldots, pa_{|P|} \rangle$, we compute its score for a query using cosine similarity metric:

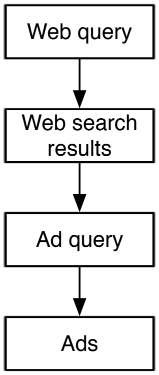

Figure 1: High-level information flow

$$score(q, a) = \alpha \sum_{i=1..|U|} uq_i \cdot ua_i + \beta \sum_{j=1..|C|} cq_j \cdot ca_j + \gamma \sum_{k=1..|P|} pq_k \cdot pa_k, \tag{1}$$

where α, β and γ are the weights reflecting the importance of the different feature classes.

Figure 2 gives an overview of the system architecture. Although we currently use three different kinds of features, our modular approach could easily incorporate additional feature types, which could be built using additional knowledge sources.

3.2 Feature construction

Our primary source of augmenting the Web query and constructing new features is the set of top-scoring search results for the original Web query. We adopt the blind relevance feedback approach, and assume that most of the top-scoring results are relevant to the query to some extent. Let $R = \{r_1, \ldots, r_{|N|}\}$ be a set of top search results.

3.2.1 Bag of words

To construct word-level unigram features U we first pool together all the individual words that occur in search results pages. Taking all the words that occur in any of the result pages would necessarily be very noisy, hence we use feature selection to represent the query only with features that are truly characteristic of it. Since we employ the blind relevance approach, we do not have any kind of labeling of search results, and hence the feature selection step should be unsupervised. Therefore, we cannot use inherently supervised methods like information gain, and resort to using metrics based on document frequency and TFIDF. It should be noted, however, that studies in (supervised) text categorization confirm that feature selection based on document frequency yields results that are on par with those based on information gain [42].

Having selected a desired number of features, we assign their values using the TFIDF scheme [39], where we use

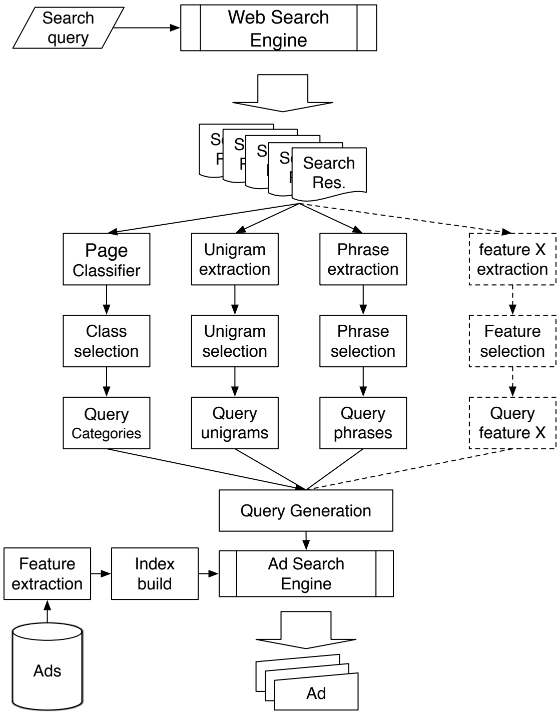

Figure 2: Architecture overview

logarithmic term frequency and IDF computed over the ad corpus. Precisely, feature weights are computed as $uq_i = (1 + log(tf))log\frac{NA}{NA(uq_i)}$, where tf is the number of occurrences of uq_i in the pooled search results $\cup_i r_i$, NA is the total number of ads, and $NA(uq_i)$ is the number of ads whose text contains the word uq_i. Finally, unigram weights undergo cosine normalization: $uq'_i = \frac{uq_i}{\sqrt{\sum_{i=1..|U|} uq_i^2}}$.

3.2.2 Query classification

If a query and an ad are highly related but use different vocabulary, the bag of words matching will be insufficient to capture their relatedness. While this problem is alleviated to some degree by expanding the query with search results, it cannot be completely solved this way. We use text classification with respect to an external taxonomy in order to identify commonalities between related but different vocabularies. To achieve this aim, we use a large taxonomy of commercial-intent topics, and build a document classifier that is capable of mapping an input fragment of text into a number of relevant classes. We then use these classes to create new features for queries and ads. Doing so not only allows us to generalize from the level of individual words to higher-level abstractions, but also explicitly benefits from the external knowledge that was used to build this auxiliary classifier.

Our choice of taxonomy was guided by a Web advertising application. Since we want the classes to be useful for matching ads, the taxonomy needs to be elaborate enough to facilitate ample classification specificity. For example, classifying all medical queries into one node will likely result in poor ad matching, as both "sore foot" and "flu" queries will end up in the same node. The ads appropriate for these two queries are, however, very different. To avoid such situations, the taxonomy needs to provide sufficient discrimination between common commercial topics. Therefore, we employed a large taxonomy of approximately $6,000$ nodes, arranged in a hierarchy with median depth 5 and maximum depth 9. Human editors populated the taxonomy with labeled bid phrases of actual ads (approx. 150 phrases per node), which were used as a training set; a small fraction of queries have been assigned to more than one category. We used the same taxonomy in our earlier work [7], where it is described in more detail.

Few machine learning algorithms can efficiently handle so many different classes and training examples. Suitable candidates include the nearest neighbor and the Naive Bayes classifier [13], as well as prototype formation methods such as Rocchio [37] or centroid-based [19] classifiers. We used the latter method to implement our text classifier. For each taxonomy node we concatenated all the phrases associated with this node into a single meta-document. We then computed a centroid for each node by summing up the TFIDF values of individual terms, and normalizing by the number of phrases in the class:

$$\vec{c_j} = \frac{1}{|C_j|}\sum_{\vec{p}\in C_j}\frac{\vec{p}}{\|\vec{p}\|},$$

where $\vec{c_j}$ is the centroid for class C_j and p iterates over the phrases in a particular class.

The classification is based on the cosine of the angle between the input document and the centroid meta-documents:

$$C_{max} = \arg\max_{C_j \in C} \frac{\vec{c_j}}{\|\vec{c_j}\|} \cdot \frac{\vec{d_j}}{\|\vec{d_j}\|}$$

$$= \arg\max_{C_j \in C} \frac{\sum_{i\in|F|} c^i \cdot d^i}{\sqrt{\sum_{i\in|F|}(c^i)^2}\sqrt{\sum_{i\in|F|}(d^i)^2}},$$

where F is the bag of words, and c^i and d^i represent the weight of the ith feature in the class centroid and the document, respectively. The scores are normalized by the document and centroid lengths to make the scores of different documents comparable.

Given the search results produced for the Web query, we classify each result page and then perform voting among them to select several classifications that best characterize the query. As reported in our previous work [7], the accuracy of this query classification approach is quite high with $Precision = 0.807$ and $F1 = 0.893$ at 100% recall. Following [17], we construct new features based on these immediate classifications as well as their ancestors in the taxonomy (the weight of each ancestor feature was decreased with a damping factor of 0.5). The weights of classification features are defined by the confidence scores assigned by the document classifier. The only transformation applied to these scores is cosine normalization.

3.2.3 Phrase extraction

For phrase extraction, we used a proprietary variant of Altavista's Prisma refinement tool [2] developed in-house. This tool includes two components, an online and an offline one. Given a fragment of text, the online component analyzes it to identify named entities and other stable phrases. This component has been integrated into the crawling and indexing pipeline of the search engine, and is routinely invoked on all the pages included in the search engine index. The offline component collectively analyzes the phrases found in all the crawled pages, and retains the most significant ones based on their statistical properties. These phrases can then be used as a restricted lexicon for indexing any piece of text they occur in. Approximately 10 million phrases (called Prisma terms in the sequel) are selected for the English language.

Given the set of search results, we first identify Prisma terms that occur in them, and then perform feature selection to retain the most characteristic ones. Both feature selection and TFIDF-based feature weighting are performed similarly to the processing of unigrams explained in Section 3.2.1. Other feature weighting schemes, notably, BM25 [36] have been reported in the literature, and we intend to report their application to sponsored search advertising in our future work.

At the end of the feature construction process we obtain an augmented query represented using three kinds of features—unigrams, classes, and Prisma terms. In contrast to a few words that comprised the original Web query, these additional features have been constructed by collectively analyzing the set of search results produced for the original Web query. The augmented ad query is then evaluated against the ad index to retrieve relevant ads.

3.3 Ad indexing and retrieval

The ads are available ahead of time and the ad processing is performed offline over the Hadoop grid-computing infrastructure (`http://lucene.apache.org/hadoop/`). We analyze the ad text and construct the same three types of features that we do for queries, namely, unigrams, classes, and Prisma terms. At this moment, we do not analyze the contents of the Web page pointed at by the ad (the landing page), as in our previous work we found that it is often too noisy.

In an online advertising system, the number of ads can easily reach tens and even hundreds of millions. Therefore, to facilitate fast ad search and retrieval we use an inverted index of ads. Finding relevant ads for the query amounts to efficiently evaluating the scores of candidate ads as defined by Equation (1), and then retrieving the desired number of highest-scoring ads.

As opposed to traditional search engines where the queries are short and documents are long, in our case ad queries are composed of Web-based features (as explained in the preceding section), and are fairly long. An ad query has on average 100–200 features, more than the number of features constructed for some ads. Therefore, we are not looking for a subsumption of the query vector by the ad vector; instead, we search for ads that are most similar to the query. To efficiently perform the similarity search over the ad space, we have adapted the WAND algorithm [8] to work with longer queries. WAND uses a branch-and-bound approach to reduce the number of ads considered. For each query feature, one cursor is opened to traverse the posting lists. The cursors are moved based on the upper bound of the score of the document that the cursor currently points at. Only documents with upper bounds higher than the minimal score in the current candidate set are considered.

3.4 Implementation efficiency

The system described in this paper used our experimental prototype where search result pages are crawled and analyzed at query time. However, in a real-life system, feature extraction can be performed at page indexing time. Since the set of possible search results is final, the search engine can analyze each Web page and classify it at the time the page is added to the search index. At query time, the search engine just needs to pass the precomputed features of top-scoring search results to the advertising subsystem. Thus, at runtime we do not have to pay the penalty of page crawling and feature extraction. This way, the approach presented in this paper can be practically implemented to conform to the ad serving latency requirements (several hundreds of milliseconds).

4. EMPIRICAL EVALUATION

We implemented our methodology for feature construction using relevance feedback in an ad matching platform named Onyx. In this section we report the results of its experimental evaluation.

4.1 Experimental methodology

We start with describing the implementation details and the datasets we used, and then proceed to presenting the results of empirical evaluation of our methodology.

4.1.1 Implementation details

Given a query, we run it through the Yahoo Web search engine, and keep the top 40 URL results (this number was empirically determined to be optimal for query classification [7]). We crawl the returned search results, tokenize their text, remove stop words, and stem the remaining words with the Porter stemmer [32]. For both unigrams and Prisma terms we selected up to 50 features of each type, while we evaluated feature selection based on document frequency (DF) as well as based on TFIDF weights. To construct classification features, we first obtained top 5 classes for each individual search result, and then performed voting to select 5 best classes for the query. We constructed features based on these 5 classes as well as their ancestors in the hierarchy. The optimal number of classes per query was obtained through validation on a held-out dataset.

We implemented the following four system settings (where α, β, γ are the relative weights for the different feature types; see Equation (1)):

Onyx1 $\alpha = 1.0, \beta = 0.5, \gamma = 0.5$, feature selection = DF

Onyx2 $\alpha = 0.5, \beta = 1.0, \gamma = 0.5$, feature selection = DF

Onyx3 $\alpha = 0.5, \beta = 0.5, \gamma = 1.0$, feature selection = DF

Onyx4 $\alpha = 1.0, \beta = 0.5, \gamma = 0.5$, feature selection = TFIDF

The rationale behind the first three settings is to emphasize different feature types. The choices of α, β, γ values in the three settings above were designed to provide good sampling of the parameter space. Given the human relevance judgments (explained in the following section), we also subsequently tuned the α, β and γ values for optimal performance. The fourth setting was used to evaluate the TFIDF formula as a feature selection metric.

Throughout the paper, most graphs only display the performance of the first system setting (Onyx 1), which we abbreviate as simply “Onyx”. The performance of all the four settings is presented in Figure 4.

4.1.2 Data description

We used a set of 700 Web queries, which has been constructed in the following way. We started with a set of all queries received by the Yahoo Web search engine during the week of July 23–29, 2007. We divided the 10 million most frequent queries into deciles by frequency, and randomly sampled 50 queries from each decile. We furthermore sampled 200 queries from the distribution tail (beyond the 10 million most frequent ones).

Each query has been matched to up to three ads using each of the above four system settings, resulting in over 9,000 query-ad pairings (some queries could only be matched to fewer than three ads). A team of six analysts, all of whom hold college degrees and have a high command of the English language, provided relevance judgments for each query-ad pair using the following scale: Perfect, Certainly Attractive, Probably Attractive, Somewhat Attractive, Probably Not Attractive, and Certainly Not Attractive.

In order to compute the standard metrics of precision and recall, we converted the above judgments to binary by considering the first four as relevant, and the rest as irrelevant. To compute precision at various levels recall, we ordered all the query-ad pairs by their scores (as assigned by Equation 1), and used a threshold to include progressively larger fractions of ads.

It should be noted that human relevance judgments are quite expensive, especially when thousands of judgments are needed. Therefore, our choices were either to judge several ads for many queries, or numerous ads for just a few queries. We adopted the former approach, as it provides a better assessment of our methodology for a realistic sample of queries of very different frequencies.

4.2 The effect of feature construction

In order to assess the value of our methodology, we compare its results with the baseline that does not use feature construction, and only uses features that are available from the query *per se*, without query augmentation using search results. Remember that the Onyx approach performs broad match, as it matches the entire ad text to the augmented query representation, rather than merely matching the ad's bid phrases to the original (unaugmented) Web query. Consequently, to make the comparison meaningful our baseline system also performs broad match, albeit without query augmentation. That is, the baseline system matches the query words to any part of the ad rather than solely to its bid phrases. Restricting the baseline to exact match only would drastically limit its coverage, making the results not comparable to those of Onyx.

In Section 4.4 we also compare Onyx performance to that of the log-based query substitution system [22] as another baseline.

Figure 3 shows the standard precision-recall tradeoff curve for Onyx setting 1. In the case of sponsored advertising, this curve is of particular importance for the following reason. Conventional information retrieval systems always produce some results if the queried collection contains documents that match some query words. Even though these documents may be irrelevant to the query, IR systems are normally expected to yield some results. However, in the case of Web search advertising, in some cases it is desirable not to show any ads. In this scenario, if no ads are relevant to the user's information need, then showing irrelevant ads should be avoided as it deteriorates user experience. Therefore, it is essential to ascertain that our method offers sufficiently high precision at low to medium coverage levels.

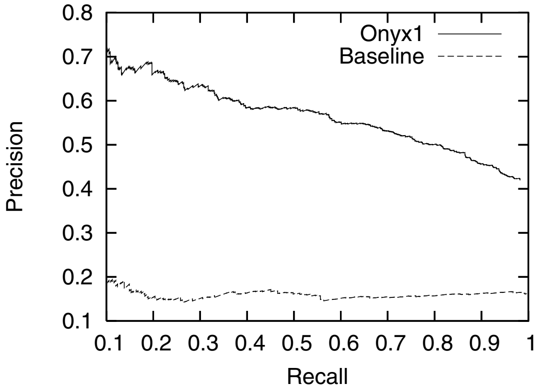

Figure 3: Precision-recall tradeoff

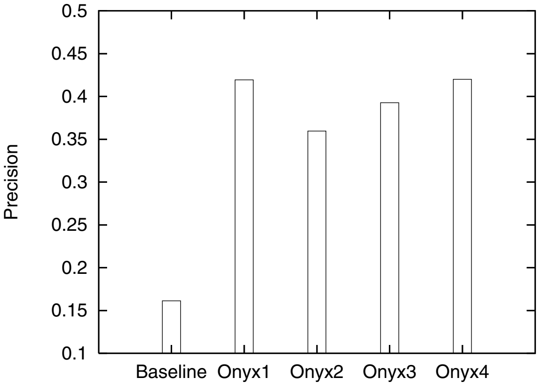

Figure 4: The effect of feature construction

Indeed, as we see in Figure 3, the precision of our method improves steadily as we reduce the fraction of queries for which ads are to be displayed. Furthermore, it can be readily seen that feature construction based on search results improves ad relevance compared to the baseline over the entire range of recall values.

In the rest of this paper, we only report Onyx precision at 100% recall (with the exception of Figure 6, see explanation in Section 4.4). However, according to the above analysis, we could always select a lower recall (and correspondingly higher precision) for actual system implementations.

Figure 4 presents the performance of the four different system settings listed in Section 4.1.1. Observe that the performance of all the four system variants is superior to that of the baseline. Interestingly, the performance of Onyx setting 4 is nearly identical to that of Onyx setting 1, implying that the quality of feature selection based on document frequency (DF) and on TFIDF is essentially the same.

We also conducted a search over all α, β, and γ combinations, with the values of each parameter varying from 0.0 to 1.0 in 0.1 increments (1,330 combinations in total, excluding the trivial combination $\alpha = \beta = \gamma = 0$). The best performance was achieved using $\alpha = 1, \beta = 0.6, \gamma = 0.4$, and it was insignificantly different from that achieved by Onyx setting 1 ($\alpha = 1, \beta = 0.5, \gamma = 0.5$).

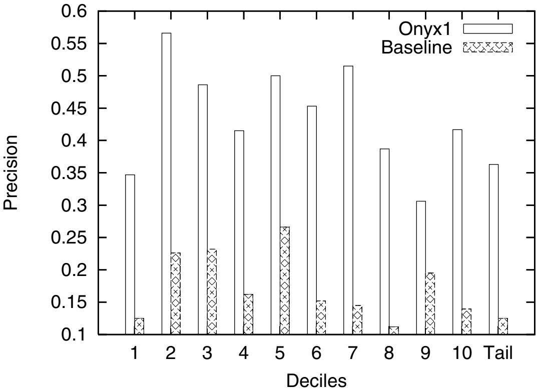

Figure 5: Onyx performance over a range of query frequencies

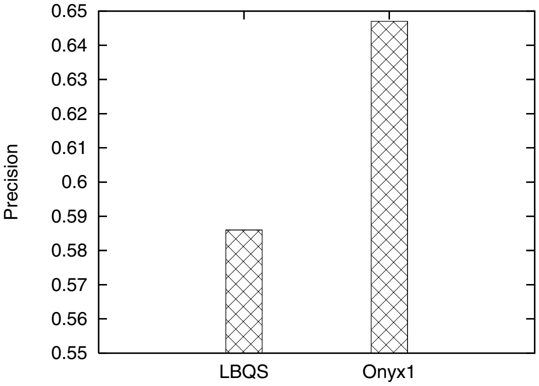

Figure 6: Comparison of Onyx and LBQS

4.3 Onyx performance for different ranges of query frequency

As explained in Section 4.1.1, we constructed our query set by stratified sampling out of a Yahoo Web search query log collected over one full week. The distribution of query frequencies follows a power law [45] (commonly referred to as Zipf's law), hence it is interesting to observe the relevance of ads that our method provides for queries of different frequencies.

Figure 5 shows Onyx and baseline performance for queries in different frequency ranges. In this figure, "1" designates the first decile (i.e., most frequent queries), "10" represents the last decile of the first 10 million queries, and "Tail" stands for rarest queries (sampled from the tail of the distribution). Observe that our methodology provides more relevant ads than the baseline for all query frequencies. Furthermore, it should be observed that our method allows to provide relevant ads even for the tail queries.

Interestingly, the relevance of ads for the most frequent queries (decile 1) is notably lower than that of less frequent queries. We believe the reason for that is that a large fraction of queries in the first decile are navigational (e.g.,"ebay", "youtube" or "hotmail"), that is the user merely wants to get the URL of the corresponding Web site. In such a case, the user is rarely interested even in search results beyond the first one, let alone any ads that might be shown, hence a majority of such ads are considered less relevant.

4.4 Comparison with log-based query substitution

As an alternative baseline, we also compare our methodology with log-based query substitution [22] (abbreviated as LBQS in the sequel). LBQS is a method designed to improve Web search queries by automatically analyzing query logs, and learning from query transformations manually performed by Web search users. Consequently, LBQS can be viewed as a query transformation technique that uses alternative source of knowledge, namely, search query logs. In the advertising scenario, we use LBQS to transform original search queries into better ones, and then match them to ads.

LBQS generates possible substitutions by first finding all pairs of successive queries issued by the same user in a search engine log, and then analyzes these queries and finds common transformations. Given a new query such as "New York maps", the system segments it into phrases using pointwise mutual information. This way, the example query would be segmented as "(New York) (maps)". To generate candidate substitutions, LBQS then applies common transformations observed earlier, for instance, transforming "maps" into "directions", yielding a substitute query "New York directions". The score of a substitution is determined by a machine learning classifier trained on a set of features that capture textual similarity as well as the frequency of the transformations applied.

Figure 6 compares the performance of LBQS and Onyx. For this experiment, we used an existing LBQS implementation that provided substitutions for 24% of the 700 queries in our dataset (for the other queries, its learned model could not apply any known transformation). Consequently, to make the comparison meaningful, Figure 6 also shows the Onyx precision at 24% level of recall.

As we can see from Figure 6, Onyx outperforms LBQS. However, it is also interesting to compare the performance of the two systems in greater depth by looking at individual queries. Table 1 shows for each of the two systems how many queries in our dataset get relevant ads, irrelevant ads, or are not covered at all. Since Onyx and LBQS use very different sources of knowledge (Web search results and search query logs, respectively), it is intuitive to understand that they perform well on very different query subsets. Furthermore, for as many as 22% of the queries, Onyx provides relevant ads while LBQS provides no ads at all. This observation implies that it is possible to design a fusion approach that provides relevant ads for an even larger fraction of input queries by using the two systems together.[2] We intend to develop such a combined approach in our future work.

LBQS:	Relevant	Irrelevant	Uncovered	Total
Onyx:				
Relevant	9.3%	3.6%	**22%**	34.9%
Irrelevant	4.7%	6.3%	48.9%	59.9%
Uncovered	0%	0%	5.2%	5.2%
Total	14.0%	9.9%	76.1%	100%

Table 1: Queries covered by LBQS and Onyx

[2] Table 1 was generated for LBQS coverage of 24% and Onyx coverage of 100%. This is meaningful for our discussion, since for higher coverage levels LBQS precision will necessarily drop, thus providing relevant ads for an even smaller fraction of the queries.

5. RELATED WORK

There have been several bodies of prior research that are relevant to our study.

5.1 Online advertising

Online advertising is an emerging area of research, so the published literature is quite sparse. A recent study [49] confirms the intuition that ads need to be relevant to the user's interest to avoid degrading the user's experience and increase the probability of reaction.

In the content match scenario, Ribeiro-Neto et al. [34] examined a number of strategies for matching pages to ads based on extracted keywords. They used the standard vector space model to represent ads and pages, and proposed a number of strategies to improve the matching process. While both pages and ads are mapped to the same space, there is a discrepancy (called "impedance mismatch") between the vocabulary used in the ads and in the pages. For example, the plain vector space model cannot easily account for synonyms, that is, it cannot easily match pages and ads that describe related topics using different vocabularies. The authors achieved improved matching precision by expanding the page vocabulary with terms from similar pages, which were weighted based on their overall similarity to the original page. In this paper, we "bridge" between related words by defining new features based on higher-level concepts from the classification taxonomy.

In their follow-up work [26], the authors proposed a method to learn the impact of individual features by using genetic programming to produce a matching function. The function is represented as a tree composed of arithmetic operators and functions as internal nodes, and different numerical features of the query and ad terms as leaves. The results show that genetic programming finds matching functions that significantly improve the matching compared to the best method (without page-side expansion) reported in [34].

Another approach to contextual advertising is to reduce it to the problem of sponsored search advertising by extracting phrases from the page and matching them to the bid phrases of the ads. Yih et al. [52] described a system for phrase extraction that uses a variety of features to determine the importance of page phrases for advertising purposes. The system is trained with pages that have been hand-annotated with important phrases. The learning algorithm takes into account features based on TFIDF, HTML meta data, and search query logs to detect the most important phrases. During evaluation, each phrase up to length 5 is considered a potential result and evaluated against the trained classifier.

Langheinrich et al. [27] studied customization techniques for matching ads to users' short-term interests. To capture short-term interests, the authors used search queries as well as visited URLs, which could then be looked up in Web directories. Jin et al. [21] used Web page classification to determine whether a given Web page does not contain sensitive content, so that it is acceptable to display ads on it.

Prior studies on sponsored search mostly experimented with the information explicitly available in the query and the ad. In contrast, in this work we study the importance of constructing new features based on exogenous sources of knowledge, such as Web search results and a large-scale taxonomy of commercial topics.

5.2 Using Web knowledge

Even though the average length of search queries is steadily increasing over time, a typical query is still shorter than 3 words. Consequently, many researchers studied possible ways to enhance queries with additional information. This can be done either using electronic dictionaries and thesauri [48], or via relevance feedback techniques that make use of a few top-scoring search results. Early work in information retrieval concentrated on manually reviewing the returned results [40, 37]. However, the sheer volume of queries nowadays does not lend itself to manual supervision, and hence subsequent works focused on *blind* relevance feedback, which basically assumes top returned results to be relevant [51, 31, 15, 35]. As an alternative to relevance feedback, other studies performed query augmentation based on the analysis of query logs [10, 22].

More recently, studies in query augmentation focused on classification of queries, assuming such classifications to be beneficial for more focused query interpretation. Indeed, Kowalczyk et al. [25] found that using query classes improved the performance of document retrieval. Studies in the field pursue different approaches for obtaining additional information about the queries. Beitzel et al. [4] used semi-supervised learning as well as unlabeled data [5]. Gravano et al. [18] classified queries with respect to geographic locality in order to determine whether their intent is local or global.

The 2005 KDD Cup on Web query classification inspired yet another line of research, which focused on enriching queries using Web search engines and directories [29, 43, 44, 23, 47]. The KDD task specification provided a small taxonomy (67 nodes) along with a set of labeled queries, and posed a challenge to use this training data to build a query classifier.

Web search results have also been used in a related task of measuring similarity of short segments of text [38, 30]. More generally, the use of search results as a source of additional features, and especially the use of Web-based Prisma terms, is also related to the studies of the Web as a corpus [24].

5.3 Cross-corpus learning

In our methodology, we use a text classifier that maps Web search results onto a taxonomy of commercial topics, whereas taxonomy nodes define new features for representing queries and ads. This approach is related to *transfer learning*, where knowledge learned in one domain is transferred to another domain. Transfer learning methods [6, 12, 46, 33] leverage information from different but related learning tasks, so that features constructed while solving one problem can be used for solving another problem.

Only a few recent studies focused on cross-corpora query augmentation. He and Peng [20], and later Diaz and Metzler [11] used several document collections to augment TREC queries. Li et al. [28] used Wikipedia as an external corpus to augment ad-hoc TREC queries. There are two notable differences between these works and our approach presented herein. First, TREC queries are usually much longer than Web queries. Second, our target collection of ads is substantially different from TREC documents, since ads are short and are often created with presentation in mind, and are hence particularly difficult for indexing. To this end, in this work we also augment indexed ads with constructed features.

6. CONCLUSIONS

Web search engines are complex systems, built as a result of many years of research and development. Running a search engine requires a sophisticated infrastructure, built and maintained to provide comprehensive and up-to-date answers to users' queries. In this work we build upon this effort by using a search engine to improve search advertising. The key idea of our approach is the use of Web search results to construct new features for the ad query, which is used to select the ads shown alongside search results. We also expand the ad representation using classification and phrase extraction.

The contributions of this paper are fourfold. First, we formulate a methodology for cross-corpora query expansion, where we use one corpus (the Web) to augment queries to be evaluated against another corpus (ads). Second, we propose a method for constructing new features based on external knowledge, which provides a richer representation of both queries and ads. Next, we relax the requirement that advertisers need to explicitly specify queries that their ads should be shown for. Instead, we use the entire contents of the ad to identify queries for which it should be shown. Using the classification-based and phrase-based features facilitates thematic matching that goes beyond the simple bag of words. The use of search results for ad matching provides an additional benefit that the search results and ads are thematically matched. Finally, we provide an evaluation of an end-to-end ad selection system based on an inverted index that supports long queries.

In our experimental evaluation we show that using the constructed features allows us to match Web search queries to significantly more relevant ads. We also conducted ablation studies to assess the individual utility of each feature class, and showed that while the bag of words is still the most important type of features for ad matching, the phrases and classes also have a significant impact on the ad selection quality. We compared our approach to a query substitution system that uses search logs as an alternative source of knowledge [22], and argued for a possibility of building a superior system by merging the two approaches.

Actual search advertising systems also incorporate past click data into the ad matching process. In this work, we focused solely on textual relevance, but in our future work we plan to combine both relevance features and click-through features. We also plan to evaluate our system in a real-world setting and measure actual click-through rates in addition to collecting human relevance judgments.

Acknowledgments

We thank Donald Metzler for fruitful discussions and pointers to relevant literature, and Bo Pang for comments on an early draft of the paper. We also thank Ann Hsieh and her editorial team for judging the ad relevance.

7. REFERENCES

[1] E. Agichtein, S. Lawrence, and L. Gravano. Learning search engine specific query transformations for question answering. In *WWW10*, 2001.

[2] P. Anick. Using terminological feedback for web search refinement: a log-based study. In *SIGIR'03*, pages 88–95, 2003.

[3] L. Ballesteros and B. Croft. Phrasal translation and query expansion techniques for cross-language information retrieval. In *SIGIR'97*, pages 84–91, 1997.

[4] S. Beitzel, E. Jensen, O. Frieder, D. Grossman, D. Lewis, A. Chowdhury, and A. Kolcz. Automatic web query classification using labeled and unlabeled training data. In *Proceedings of SIGIR'05*, pages 581–582, New York, NY, USA, 2005. ACM Press.

[5] S. Beitzel, E. Jensen, O. Frieder, D. Lewis, A. Chowdhury, and A. Kolcz. Improving automatic query classification via semi-supervised learning. In *ICDM'05*, Washington, DC, USA, November 2005. IEEE Computer Society.

[6] P. N. Bennett, S. T. Dumais, and E. Horvitz. Inductive transfer for text classification using generalized reliability indicators. In *Proceedings of the ICML-2003 Workshop on The Continuum from Labeled to Unlabeled Data*, 2003.

[7] A. Broder, M. Fontoura, E. Gabrilovich, A. Joshi, V. Josifovski, and T. Zhang. Robust classification of rare queries using web knowledge. In *SIGIR'07*, 2007.

[8] A. Z. Broder, D. Carmel, M. Herscovici, A. Soffer, and J. Zien. Efficient query evaluation using a two-level retrieval process. In *CIKM'03*, pages 426–434, 2003.

[9] 61 billion searches conducted worldwide in August. comscore, October 2007. Available from `http://www.comscore.com/press/release.asp?press=1802`.

[10] H. Cui, J.-r. Wen, J.-Y. Nie, and W.-Y. Ma. Probabilistic query expansion using query logs. In *WWW'02*, 2002.

[11] F. Diaz and D. Metzler. Improving the estimation of relevance models using large external corpora. In *SIGIR'06*, 2006.

[12] C. Do and A. Ng. Transfer learning for text classification. In *Proceedings of Neural Information Processing Systems (NIPS)*, 2005.

[13] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons, 1973.

[14] B. Edelman, M. Ostrovsky, and M. Schwarz. Internet advertising and the generalized second price auction: Selling billions of dollars worth of keywords. *American Economic Review*, 97(1):242–259, 2007.

[15] E. Efthimiadis and P. Biron. UCLA-okapi at TREC-2: Query expansion experiments. In *Proceedings of TREC-2*, 1994.

[16] D. Fain and J. Pedersen. Sponsored search: A brief history. In *Second Workshop on Sponsored Search Auctions*, 2006.

[17] E. Gabrilovich and S. Markovitch. Feature generation for text categorization using world knowledge. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, pages 1048–1053, Edinburgh, Scotand, August 2005.

[18] L. Gravano, V. Hatzivassiloglou, and R. Lichtenstein. Categorizing web queries according to geographical locality. In *CIKM'03*, pages 325–333, 2003.

[19] E.-H. S. Han and G. Karypis. Centroid-based document classification: Analysis and experimental results. In *Proceedings of the Fourth European Conference on Principles and Practice of Knowledge Discovery in Databases*, September 2000.

[20] D. He and Y. Peng. Comparing two blind relevance feedback techniques. In *SIGIR'05*, pages 649–650, 2005.

[21] X. Jin, Y. Li, T. Mah, and J. Tong. Sensitive webpage classification for content advertising. In *Workshop on Data Mining and Audience Intelligence for Advertising (ADKDD) at KDD'07*, pages 28–33, August 2007.

[22] R. Jones, B. Rey, O. Madani, and W. Greiner. Generating query substitutions. In *WWW'06*, pages 387–396, May 2006.

[23] Z. Kardkovacs, D. Tikk, and Z. Bansaghi. The ferrety algorithm for the KDD Cup 2005 problem. In *SIGKDD Explorations*, volume 7, pages 111–116. ACM, December 2005.

[24] A. Kilgariff and g. Grefenstette. Introduction to the special issue on the web as corpus. *Computational Linguistics*, 29(1):333–347, 2003.

[25] P. Kowalczyk, I. Zukerman, and M. Niemann. Analyzing the effect of query class on document retrieval performance. In *Proceedings of the Australian Conference on Artificial Intelligence*, pages 550–561, 2004.

[26] A. Lacerda, M. Cristo, M. A. Goncalves, W. Fan, N. Ziviani, and B. Ribeiro-Neto. Learning to advertise. In *SIGIR'06*, pages 549–556, 2006.

[27] M. Langheinrich, A. Nakamura, N. Abe, T. Kamba, and Y. Koseki. Unintrusive customization techniques for web advertising. *Computer Networks*, 31:1259–1272, May 1999.

[28] Y. Li, R. Luk, E. Ho, and F. Chung. Improving weak ad-hoc queries using wikipedia as external corpus. In *SIGIR'07*, pages 797–798, 2007.

[29] Y. Li, Z. Zheng, and H. Dai. KDD CUP-2005 report: Facing a great challenge. In *SIGKDD Explorations*, volume 7, pages 91–99. ACM, December 2005.

[30] D. Metzler, S. Dumais, and C. Meek. Similarity measures for short segments of text. In *Proceedings of the 29th European Conference on Information Retrieval*, pages 16–27, 2007.

[31] M. Mitra, A. Singhal, and C. Buckley. Improving automatic query expansion. In *SIGIR'98*, 1998.

[32] M. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[33] R. Raina, A. Ng, and D. Koller. Constructing informative priors using transfer learning. In *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, Pittsburgh, PA, 2006.

[34] B. Ribeiro-Neto, M. Cristo, P. B. Golgher, and E. S. de Moura. Impedance coupling in content-targeted advertising. In *SIGIR'05*, 2005.

[35] S. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In *Proceedings of TREC-3*, 1995.

[36] S. E. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. okapi at TREC-3. In *Proceedings of the 3rd Text REtrieval Conference*, 1994.

[37] J. J. Rocchio. Relevance feedback in information retrieval. In *The SMART Retrieval System: Experiments in Automatic Document Processing*, pages 313–323. Prentice Hall, 1971.

[38] M. Sahami and T. Heilman. A web-based kernel function for measuring the similarity of short text snippets. In *WWW'06*, pages 377–386. ACM Press, May 2006.

[39] G. Salton and C. Buckley. Term weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988.

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

[41] G. Salton and M. McGill. *An Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.

[42] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[43] D. Shen, R. Pan, J. Sun, J. Pan, K. Wu, J. Yin, and Q. Yang. Q2C@UST: Our winning solution to query classification in KDDCUP 2005. In *SIGKDD Explorations*, volume 7, pages 100–110. ACM, December 2005.

[44] D. Shen, J. Sun, Q. Yang, and Z. Chen. Building bridges for web query classification. In *SIGIR'06*, pages 131–138, New York, NY, USA, 2006. ACM Press.

[45] A. Spink, D. Wolfram, B. Jansen, and T. Saracevic. Searching the web: The public and their queries. *Journal of the American Society for Information Science and Technology*, 53(3):226–234, 2001.

[46] C. Sutton and A. McCallum. Composition of conditional random fields for transfer learning. In *Emprical Methods in Natural Language Processing (HLT/EMNLP)*, 1998.

[47] D. Vogel, S. Bickel, P. Haider, R. Schimpfky, P. Siemen, S. Bridges, and T. Scheffer. Classifying search engine queries using the web as background knowledge. *SIGKDD Explorations*, 7(2):117–122, 2005.

[48] E. M. Voorhees. Query expansion using lexical-semantic relations. In *Proceedings of the 17th International Conference on Research and Development in Information Retrieval*, pages 61–69, 1994.

[49] C. Wang, P. Zhang, R. Choi, and M. D. Eredita. Understanding consumers attitude toward advertising. In *8th Americas Conference on Information Systems*, 2002.

[50] J. Xu and W. B. Croft. Query expansion using local and global document analysis. In *Proceedings of the 19th International Conference on Research and Development in Information Retrieval*, pages 4–11, 1996.

[51] J. Xu and W. B. Croft. Improving the effectiveness of information retrieval with local context analysis. *ACM TOIS*, 18(1):79–112, 2000.

[52] W.-t. Yih, J. Goodman, and V. R. Carvalho. Finding advertising keywords on web pages. In *WWW'06*, 2006.

[53] J. Zobel and A. Moffat. Exploring the similarity space. *ACM SIGIR Forum*, 32(1):18–34, 1998.