Optimizing Relevance and Revenue in Ad Search: A Query Substitution Approach

Filip Radlinski[‡1], Andrei Broder[†], Peter Ciccolo[†], Evgeniy Gabrilovich[†], Vanja Josifovski[†], Lance Riedel[†]

‡ Department of Computer Science, Cornell University, Ithaca, NY 14853, USA
† Yahoo! Research, 2821 Mission College Blvd, Santa Clara, CA 95054, USA

filip@cs.cornell.edu | {broder | ciccolo | gabr | vanjaj | riedell }@yahoo-inc.com

ABSTRACT

The primary business model behind Web search is based on textual advertising, where contextually relevant ads are displayed alongside search results. We address the problem of selecting these ads so that they are both relevant to the queries and profitable to the search engine, showing that optimizing ad relevance and revenue is not equivalent. Selecting the best ads that satisfy these constraints also naturally incurs high computational costs, and time constraints can lead to reduced relevance and profitability. We propose a novel two-stage approach, which conducts most of the analysis ahead of time. An offline preprocessing phase leverages additional knowledge that is impractical to use in real time, and rewrites frequent queries in a way that subsequently facilitates fast and accurate online matching. Empirical evaluation shows that our method optimized for relevance matches a state-of-the-art method while improving expected revenue. When optimizing for revenue, we see even more substantial improvements in expected revenue.

Categories and Subject Descriptors

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

General Terms: Algorithms, Economics, Performance

Keywords: Online advertising, Relevance, Revenue

1. INTRODUCTION

Online advertising has arguably become the financial backbone of today's Internet. It is a multi-billion dollar market that brings audience and potential customers to countless Web sites, and provides substantial revenue for search engines and numerous content providers. The bulk of ads today are in textual form, and are selected to be *contextually* relevant to actual Web content. The two main types of textual advertising are *sponsored search*, where paid ads are displayed alongside Web search results, and *content match*, where ads are displayed on third-party Web sites. Previous studies have shown that more relevant ads lead to improved user satisfaction and higher response rates [5, 20].

We focus on improving the relevance of ads in sponsored search while optimizing the revenue they yield. The challenge in selecting relevant ads stems from the difficulty to adequately represent queries and ads in a sufficiently rich feature space. Web search queries are on the average only about 2.5 words long, and hence often difficult to interpret automatically. Textual ads are usually created with the presentation rather than indexing needs in mind, and consist of only a few dozen words. To make ad placement easier, most search engines allow advertisers to annotate ads with designated queries for which the ad is to be displayed. These queries are called *bid phrases* because advertisers participate in auctions where they bid to advertise on the designated queries. By default, an *exact match* between an ad's bid phrase and the user's query is required to display the ad. However, it can be quite difficult for advertisers to come up with an exhaustive list of bid phrases pertinent to their offers. To circumvent this difficulty, advertisers often opt-in for *broad match*, where ads can be shown for queries that do not exactly match the bid phrase, but rather are related to it. Relaxing the requirement for an exact match does not, however, supply any additional information, and the broad match problem remains just as hard.

With such short text fragments, it is essential to use additional sources of information for ad selection, and indeed prior studies employed text classification and named entity extraction [2, 3]. However, the amount of external knowledge and computation one can use is bounded by the need to provide sub-second response times. But what if we could selectively spend more time on certain queries?

We propose a two-stage approach that fixes a large set of sufficiently frequent queries, and analyzes them in depth ahead of time in a preprocessing phase. To facilitate fast online matching, the preprocessing phase constructs a lookup table that maps queries into bid phrases of appropriate ads. This step effectively transforms users' queries into alternative queries that are better for selecting ads. In the subsequent online phase, suitable ads are retrieved nearly instantaneously by exact match on the substituted query. It is entirely practical to compute such a transformation table for millions of frequent and highly monetizable queries, while the process can be repeated periodically to follow the changes in query popularity and ad supply over time.

During preprocessing, we first use pseudo-relevance feedback to expand the query representation with Web search results, and then identify a set of candidate ads. These ads provide a set of candidate bid phrases that are relevant

[1]Research performed during internship at Yahoo! Research.

SIGIR'08, July 20–24, 2008, Singapore.

to the query. We measure the quality of these bid phrases using several sources of additional knowledge, such as frequency statistics, textual similarity with respect to an external taxonomy, and bid amounts of candidate ads. It is through this use of external knowledge without impending time constraints that we achieve improved ad revenue while maintaining high relevance.

The contributions of this paper are threefold. Our main finding is that optimizing relevance alone is not sufficient for ad matching. Prior work (notably, [12]) often produced substitute queries that were not monetizable, because no or few advertisers bid on them. Using external knowledge in the offline preprocessing phase, we achieve a substantial improvement in expected advertising revenue without sacrificing the relevance of ads. The use of broad match also allows us to leverage market inefficiencies, which occur when substantially equivalent queries solicit very different bid amounts. Second, we are able to find relevant ads for a larger fraction of queries than previous techniques. We demonstrate that the relevance region of our method differs from that of the previous state-of-the-art method [12], thus making it possible to design a fusion approach that provides highly relevant substitutions for an even larger fraction of input queries. Finally, our methodology combines the flexibility of broad match and the computational efficiency of exact match ad retrieval. In essence, we show how to transform any query-to-ad matching system into an ahead-of-time query-to-query substitution system, simultaneously making use of heterogenous sources of external knowledge.

2. BACKGROUND

This section presents more details on online textual advertising, followed by a brief background on query substitution.

2.1 Online Textual Advertising

In the sponsored search scenario, online textual ads are shown next to Web search results. Usually, an ad consists of a title (typically 3–5 words long), a short description (around 20 words), and a landing URL that users who click on the ad are redirected to. Each ad is also associated with one or more bid phrases, which are Web queries (typically 2–3 words long) that advertisers list as relevant to the ad. Each bid phrase is associated with a bid amount that indicates the maximum amount of money the advertiser is willing to pay for each click on the ad. Given a query, ads are usually selected by one of the following two approaches. Exact match selects ads whose bid phrase matches the query exactly. Broad match attempts to find ads that match the user's intent expressed by the query.

In most online advertising systems, advertisers pay the search engine every time their ad is clicked on by a user. The amount paid is determined by an auction, although the details of this auction are beyond the scope of the paper. We refer interested readers to [8].

With exact match, there are three main reasons why a query q may have poor advertising potential. First, for sufficiently rare queries, there may be no ads with this exact bid phrase. Second, there may only be one or two advertisers bidding on q. Even if the ads are relevant, the standard auction approach means that the search engine is likely to receive little revenue in the event of a click. Third, the query may be ambiguous, so that many different types of ads are possible and most ads shown are irrelevant. Our method addresses the first two reasons for poor advertising performance of the exact match method.

Algorithm 1 Constructing the query substitution table

Input: Queries Q; Advertisement corpus Ads;
Result size S; Pool size k; weight vector $\vec{w}$.
for $q \in Q$ **do**
  $R_q \leftarrow TopSWebSearchResults(q, S)$
  $AdPool \leftarrow k$ ads with highest $similarity(ad, R_q)$
  $BidPhrasePool \leftarrow \{bidPhrase(ad) \mid ad \in AdPool\}$
  for $bp \in BidPhrasePool$ **do**
    $\vec{f}_{bp} \leftarrow queryFeatures(q, bp)$
  end for
  $sub(q) \leftarrow \text{argmax}_{bp \in BidPhrasePool}\, \vec{w} \cdot \vec{f}_{bp}$
end for

2.2 Query Substitution

The goal of query substitution is, given a query q, to find a substitute query $sub_{\mathcal{C}}(q)$ that provides better search results than q on corpus $\mathcal{C}$. For example, when searching for "dog diseases" in a medical corpus, the query "canine diseases" may be more effective. In general, the substitute query may involve adding or removing terms from q, or possibly transforming or replacing individual words or the whole query. Note that in our case two corpora are involved (an ad corpus and a Web corpus), so the vocabulary used for ad search ($sub_{\mathcal{C}}(q)$) need not be the same as that for general Web search (q), and can instead be optimized for retrieval on the ad corpus $\mathcal{C}$.

Although there has been a vast amount of research on query substitution, we limit ourselves to a brief overview so as to place our work in context. At a high level, queries are typically either changed incrementally from the input query or transformed into an entirely different query. An example of incremental change is stemming, which removes the endings of query words, so that the query "cameras" would match documents with the word "camera" [15]. Other examples of incremental changes include removing one or more query words to improve recall [11], correcting spelling [6], or using a dictionary and thesaurus to find synonyms of query words [19]. Common methods for transforming queries completely include Latent Semantic Indexing (LSI) [7], which maps all queries and documents to a new feature space, and building language models [14, 22] of relevant documents for example using pseudo-relevance feedback [16, 17, 21].

Many of these techniques are commonly used by Web search engines. However, both types of transformations often produce queries that are not directly useful for ad selection by exact match. For example, stemming often results in incomplete words and does not work for product names and numbers. Language modeling approaches tend to generate long queries that are unlikely to be bid on.

3. METHODOLOGY

Our approach is a hybrid of exact match and broad match. In the offline phase, we fix a large set of sufficiently frequent queries, and learn a function that substitutes input queries with one or more alternatives. Then, in the online phase, we use exact match to find ads matching the substitute query. Algorithm 1 describes the offline computations for creating our query substitution table. Given a set of queries and a corpus of ads, we iterate over all the queries. First, for each query we obtain the top S results returned by a Web search engine. Using features of these Web search results (see Sec-

tion 3.1) we find the k ads most related to the input query. The bid phrases of these ads are taken as a candidate pool of possible substitutions for the query. For every candidate substitution, we compute a linear combination of features that measure the match between the query and bid phrase (Section 3.2). The highest scoring bid phrase is selected as the substitution for the query.

The most related previous work is log-based substitution by Jones et al. [12], which generates substitutions using search engine query logs. The method first identifies all pairs of successive queries issued by the same user, and analyzes them to find common transformations. Then, given a new query such as "new york maps", it is segmented into phrases such as "(new york) (maps)". To generate substitutions, common transformations observed earlier are then applied, for example replacing "maps" with "directions", yielding a substitute query "new york directions". This approach has several limitations. First, most of the rewrites found are not actual bid phrases and cannot be used for ad placement. Second, many queries do not have relevant transformations in the search logs. For instance, queries consisting of product codes fall into this category. This lack of data reduces the fraction of queries to which log-based substitution can be applied to around 50%. Finally, when suitable substitutions are found, these can select ads with lower revenue potential, as we will see in the evaluation.

3.1 Candidate Selection

We now describe our approach in greater detail. Given an input query, the first stage in our algorithm is to find a set of possible query substitutions. In this section, we describe this process, which we call candidate selection. We start by presenting an intuitive technique similar to Web search, followed by an enhanced version that we found slightly more effective. The similarity metrics presented here were those we found to work best, although due to space constraints we cannot discuss alternatives. A full analysis of candidate selection is in preparation [4].

3.1.1 Simple Pooling

Our first approach for generating candidate substitutions involves query expansion using a Web search engine. Each of the top S search results for the input query was downloaded and represented as a bag of words. After stop word removal and stemming, we selected the F most frequent words in the top results, and weighted each word using a variant of TFIDF weighting [18], $(1+\log(TF))\times log(N/N_d)$, where TF is the number of occurrences of the word in the top results, N is the total number of ads in our corpus and N_d is the number of those that contain the word. The specific number of search results ($S = 50$) and words ($F = 50$) were chosen using a validation dataset.

We performed the same word transformations on the ad corpus, which consists of a several hundred million actual ads from a major search engine. It is common for one ad to be associated with many different bid phrases, hence we replicate these ads so that each ad in the corpus has exactly one bid phrase. For each ad, we used the words in the title, description, bid phrase, and URL.

Given the representation of queries and ads as a weighted bag of words, we used cosine similarity to find 100 ads closest to the query. We collected the bid phrases of these ads to create the candidate pool. Note that the same bid phrase was often associated with several ads from the top 100, leading to a typical pool size of 20 candidate substitutions.

3.1.2 Enhanced Pooling

In addition to using weighted words and cosine similarity to select the top 100 ads, we also added two other candidate similarity measures to improve the quality of the ad pool.

To select words to represent queries and ads that are more informative than highly weighted words in search engine results, we used a variant of the Prisma tool [1] to generate related phrases for both. Given a text fragment, this tool extracts named entities and common phrases, and filters them through a restricted lexicon of about 10 million phrases identified through global analysis of a Web corpus. We applied Prisma on the ad text "as-is", but queries were expanded using top Web search results as explained in Section 3.1.1. We selected the 50 highest scoring phrases as features to describe each query and ad. Computing the cosine similarity of these vectors provided a second measure of similarity.

Furthermore, previous work has shown that IR performance can be improved if queries and documents are classified in a topic hierarchy [13, 9]. We classified queries and ads with respect to a taxonomy of over 6,000 commercial topics, arranged in a hierarchy with a median depth of 5. Following Broder et al. [2], we augmented the query representation with top search results, and then performed voting to obtain 5 top-scoring classes for each query; a similar process was used to classify ads. The similarity of the classification of the query and ad that Broder et al. describe served as a third similarity measure.

The final similarity of an ad to the input query is a weighted sum of the three measures, weighting the web similarity twice as much as the two new measures.

3.2 Ranking Candidate Substitutions

The simplest way to rank the candidate query substitutions is using the similarity score described above. However, such a method would completely ignore the bid information associated with each of the candidates, which is essential to optimize revenue. For example, the top ranked substitution may only have one bidding ad. Additionally, noisy matches are possible, for example selecting one candidate that may be very different from all others. Therefore, although we can consider the similarity between the query and ads as one feature, it is likely that additional features can be used to improve the quality of the query substitution found.

For these reasons, we rerank candidate substitutions using a variety of features that characterize the original query and the candidate rewrites. We use a regression support vector machine [10] with default settings to learn the weights for these features, which then allows us to compute the final score for each candidate. We then take the highest scoring candidate as the selected substitution for the input query.

3.2.1 Query Substitution Features

To describe the quality of match between a query and candidate substitution, we use three types of features motivated by three desirable properties. First, substitutions should be somehow similar to the original query, both in terms of lexical similarity (as was found by Jones et al. [12]) and semantic similarity. The latter can be measured, for example, by whether the original query and candidate substitution return similar Web search results. Second, the substitution eventually selected for a query should be representative of many candidate substitutions so as to ensure it is not a noisy match. Third, motivated by the advertising application, a good substitution should retrieve profitable ads.

We use the following features to quantify the lexical similarity between a query q and a substitution candidate t:

- *shareWords(q,t)*: Do q and t share any words?
- *wordDistance(q,t)*: The number of word changes between q and t.
- *cosine(q,t)*: The cosine similarity of q and t.
- *editDistance(q,t)*: The number of character changes between q and t.
- *trigramCosine(q,t)*: The cosine similarity of q and t if we remove all whitespace then count all three-letter subsequences.

For features that are not guaranteed to be in the interval $[0, 1]$, we first ranked all candidate substitutions by the raw feature value, and then converted the feature values to percentile ranks. For example, if 40% of the candidates for a given query had an edit distance above 10, then a candidate with edit distance 10 from the input query would have the *editDistance* equal to 0.4. We used such a transformation for all the features presented in this section.

To measure semantic similarity between a query and substitution, we again used a Web search engine. Intuitively, the "semantic" features capture the Web search engine's ability to retrieve relevant Web results. We use top Web results as background knowledge, and construct a set of features that encode semantic meaning rather than mere textual similarity measured by the lexical features:

- *maxMatchScore(q,t)*: The maximum similarity score (as described in Section 3.1) between q and any advertisement in the corpus with the bid phrase t.
- *abstractCosine(q,t)*: The cosine similarity of Q and T, where Q is the concatenation of the abstracts of the top 40 search results for q, and T is that of the abstracts of the top 40 search results for t.
- *taxonomySimilarity(q,t)*: The similarity of q to t with respect to the abovementioned classification taxonomy.

Note that the second and third features are very similar to two of the similarity measures used in the enhanced pooling approach (Section 3.1.2). However, here the similarity is measured between the query and a candidate bid phrase, rather than between the query and an entire ad as before.

We would also like the top ranked substitution to be representative of the entire pool of candidate substitutions. This way, a particular substitution is less likely to be selected based on one spurious match. This is motivated by query expansion using pseudo-relevance feedback, where it is usually optimal to add words common to many of the top ranked results retrieved for the original query. We used the following features to capture whether many of the substitution candidates are of interest to the same advertisers, and whether the substitutions are often used by search engine users:

- *clientCosine(clients(top-10),clients(t))*: The overlap between the advertisers who bid on t, and those that bid on any of the top 10 candidate substitutions in the pool.
- *queryFrequency(t)*: The frequency with which the substitution was used as a query in a major search engine.

Finally, motivated by the goal of maximizing revenue for the search engine, we used the following features to characterize the revenue potential of a given substitution t:

- *maxBid(t)*: The max bid of any ad with bid phrase t.
- *secondBid(t)*: The second highest bid for this substitution. Since search engines often use second-bid auctions to determine the actual click price, this feature can be seen as a proxy for expected revenue from a click.

Precise (1.0)
No change in meaning. Little or no change in scope.
Approximate (0.5)
Modest change in intent. The scope may have expanded or narrowed.
Marginal (0.2)
Shift in user intent to a related, but distinct topic.
Mismatch (0.0)
The original user intent has been lost or obscured.

Table 1: Relevance scale used by human judges to evaluate the match between a query and substitute.

Certainly Attractive (1.0)
Strongly related to the commercial intent or explicit need of the query; Very attractive to a user.
Probably Attractive (0.5)
Ad is slightly off-target while at the same time remaining an interesting and relevant ad to the user.
Somewhat Attractive (0.4)
Ad is perhaps not what the user originally intended, but the user still might click on it.
Probably Not Attractive (0.2)
Ad is a significant departure from the intent of the original query, yet there is still a relationship.
Certainly Not Attractive (0.0)
Ad is not related to the intent of the query.

Table 2: Relevance scale used by human judges to evaluate the match between a query and an ad.

3.2.2 Learning to Rank Candidate Substitutions

The features described above measure the match between a query and a candidate substitution. To learn a query substitution function, we need to combine these features into a final score that can be used to select a specific substitution from the candidate pool for a given query.

We learned a weighted linear combination of the features using labeled training data obtained from expert judgments. Due to the time and expense necessary to obtain these judgments, we started with a subsample of 40 diverse Web search queries used during an earlier study [12]. For these queries, we generated candidate substitutions as described above using both the simple and enhanced pooling techniques. The candidates were ranked by *maxMatchScore*, using an unweighted sum of all the features above and by using log-based substitution. The top three substitutions generated by each method were collected, giving us about 100 distinct (query, substitute) pairs after removing duplicates.

The relevance of these substitutions was evaluated by expert human judges. Judges were asked to score each (query, substitute) pair on the 4-point relevance scale shown in Table 1. The scale is designed to be granular enough to capture different levels of relevance, while restricted to allow judgments to be reliable. In addition, each level is assigned a numeric score that measures the relative relevance that the judgment represents. By training a regression SVM [10] with default settings to the relevance values, we obtained a weighted linear combination of all the features, giving us a function for ranking the candidate query substitutions.

3.2.3 Optimizing for Revenue

The ranking function learned using relevance judgments can only be expected to rank substitutions in terms of relevance to the original query. However, as described earlier, the substitutions selected should also produce high revenue for the search engine.

We propose to rank candidate substitutions by the product of (1) the query substitution relevance score and (2)

second highest bid amount of any ad bidding on the candidate substitution. Second price auctions are standard in ad pricing, hence the second highest bid amount is the revenue that a search engine would collect for a click on the top ad. We call this method *revenue optimized*.

Algorithmically, revenue optimization amends line 9 of Algorithm 1, replacing $\vec{w} \cdot \vec{f}_{bp}$ with $\vec{w} \cdot \vec{f}_{bp} \times secondBidAmount(bp)$. The relevance score $\vec{w} \cdot \vec{f}_{bp}$ is assumed to be proportional to the probability that a user will click on an ad for the substitute query after having issued the search query. Multiplying the score by the second-highest bid amount yields an estimate of the expected revenue from this query substitution.

4. EMPIRICAL EVALUATION

In this section we evaluate the performance of our approach. We start by describing the corpus of query and ad relevance judgments used for evaluation. Then we compare the expected revenue generated by using our system to previous work. We follow this by evaluating the contribution of the individual components of our technique to the overall result. Finally, we analyze the weights for the features used to rank candidate substitutions.

4.1 Evaluation Corpus

We evaluated our approach using two kinds of expert judgments, which captured the quality of query substitutions directly, as well as indirectly by evaluating the actual ads retrieved through the use of substitutions.

To build the first evaluation dataset (to measure query substitution quality), we collected a list of the 10 million most frequent queries issued to the Web search engine over a one week period. We used stratified sampling to randomly select 50 queries from each decile in terms of query frequency, giving us 500 evaluation queries; this set contained both very frequent and less frequent queries. For each evaluation query, we computed the top 3 substitutions using a variety of methods to allow us to compare their performance. The methods we used were (1) log-based substitution; (2) the learned ranking function using both the simple and enhanced pooling methods; (3) ranking substitutions by the estimated relevance times the second highest bid amount for this substitution; (4) ranking the candidate substitutions by the *maxMatchScore* feature, both using simple pooling and enhanced pooling. In the enhanced pooling case, we also multiplied the match score by the second highest bid amount. After removing duplicates, this gave us over 5,400 (query, substitute) pairs, which were judged in the same way as before.

To allow us to evaluate the relevance of the actual ads that would be returned by these methods, we also collected a second set of relevance judgments. Taking the highest scoring substitute query returned by each of the methods listed above, we collected the three ads in our corpus with highest bids. Accounting for duplicates and substitutions with fewer than three ads, this gave us almost 4,000 (query, ad) pairs. The relevance of these ads to the original query was evaluated by the expert judges, using the scale described in Table 2.

4.2 Revenue Performance

Figure 1 plots a score that is proportional to the expected revenue as a function of coverage. We calculated expected revenue as the product of the (judged) relevance score for query substitutions and the second highest bid, averaged

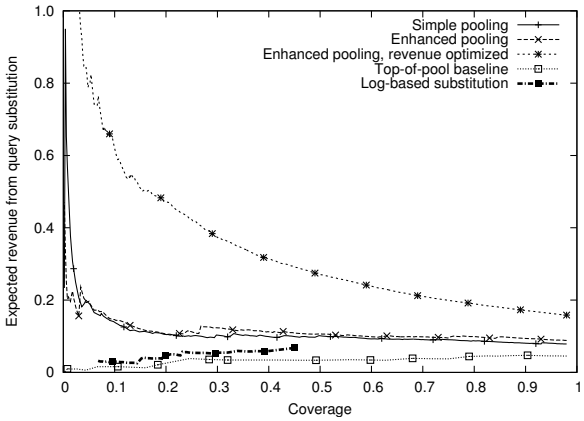

Figure 1: Expected revenue vs. query coverage

across all the evaluation queries. Along the horizontal axis, coverage indicates the fraction of the 500 evaluation queries for which each method generated a query substitution. To control the coverage of each method, we only used substitutions for which the score was above a threshold. For instance, a coverage of 40% indicates that substitutions are accepted for the 40% of the input queries with highest score. Comparing performance at different coverage levels is similar to comparing the performance of information retrieval systems at different recall levels.

Assuming that users' click probabilities are proportional to the scores assigned by the human relevance judges, the revenue optimized method (which uses enhanced pooling) generates substantially more revenue than any of the other approaches. Without revenue optimization, simple pooling and enhanced pooling perform almost indistinguishably and with much lower expected revenue. The log-based substitution technique of Jones et al. yields substantially lower expected revenue still. For comparison, we used a baseline that substitutes the input query with the top ranked bid phrase returned by simple pooling.

In the figure, we only present the curve for log-based substitutions for a limited coverage range as we were unable to practically obtain query substitutions for higher coverage levels. The maximum level of coverage shown here for that approach is similar to that reported by Jones et al. We were also unable to create a revenue-optimized version of log-based substitution because we were unable to obtain a sufficiently large selection of candidate substitutions to rank for each input query.

It is interesting to note that both the baseline and log-based substitution methods perform particularly poorly at low coverage. This suggests that the highest scoring substitutions generated, i.e., those with the highest relevance score, are rewrites that provide low revenue.

4.3 Relevance Performance

We now evaluate the performance of our approach in terms of the relevance of the ads and query substitutions found.

4.3.1 Advertisement Relevance

We start by analyzing the judged relevance of the ads selected by each method. Figure 2 presents the fraction of top 3 ads returned for the 500 evaluation queries that were relevant to the original query (i.e., judged as certainly attractive or probably attractive), as a function of the fraction

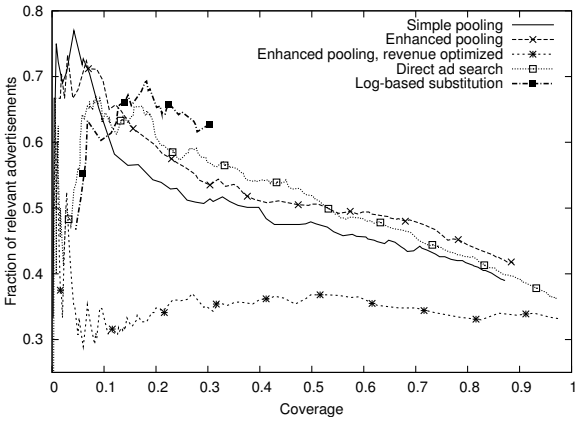

Figure 2: Fraction of relevant ads vs. query coverage.

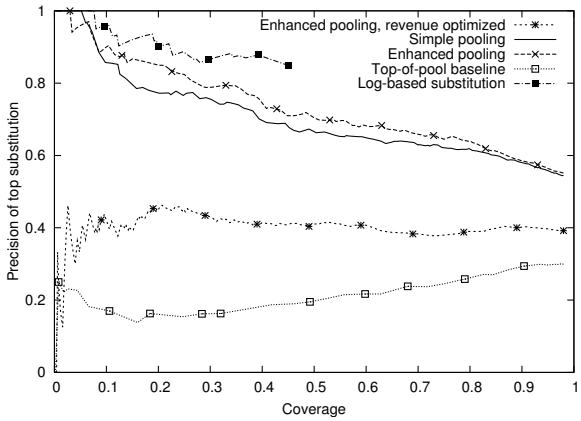

Figure 3: Fraction of query substitutions judged precise or approximate vs. query coverage.

of queries for which we generate ads. In this case we varied the coverage by choosing not to display ads when the score from each system fell below varying thresholds.

The figure shows the performance of the methods described above, and also compares to the relevance of the top ads selected by searching the ad corpus directly using enhanced pooling, without using query substitution. We see that query substitution with enhanced pooling selects more relevant ads than simple pooling, and yields ads of similar relevance to direct ad search. We also see that at high coverage levels enhanced pooling provides the most relevant ads, suggesting that query substitution helps avoid noisy ad matches. However, between 10% and 30% coverage, log-based substitution yields the most relevant ads. This happens since log-based substitution selects substitutions just based on similarity to the original query. Yet as before, log-based substitution provides substitutions for only half the queries. In addition, some of the substitute queries have no advertisements in our corpus. We also see that, surprisingly, at lowest coverage log-based substitution and direct ad search both return less relevant ads.

Note also that the revenue optimized method, which performed by far the best when measuring revenue, performs less well in terms of ad relevance. On average, the ads returned are less relevant than those returned by the other methods. This happens because the bid phrases selected as substitutions are biased toward ads that have high bid amounts. Should we wish to avoid showing ads with low relevance values, a potential approach would be to interpolate between the revenue optimized approach and another approach. For example, candidate query substitutions could be ranked by a weighted sum of the expected revenue and the ad relevance. By tuning the relative weighting, we could trade off mean relevance and expected revenue. Combined with the previous figure, these results show that optimizing for revenue and optimizing for relevance can lead to substantially different learned query substitutions. In fact, it is apparent that optimizing for revenue and optimizing for relevance can be at odds with each other, where as one improves the other decreases.

4.3.2 Query Substitution Relevance

When using query substitution for ad selection, less relevant ads may be returned for one of two reasons. Either, the learned substitution may not be relevant to the original query, or the substitute query may not have any relevant ads in our ad corpus. To tease apart the extent to which each of these two reasons contribute to the results seen in Figure 2, Figure 3 shows how the fraction of substitutions that are relevant changes as a function of query coverage. We consider substitutions judged to be a precise match or an approximate match as relevant. Log-based substitution obtains the most relevant substitutions although again only for about half the queries. Enhanced pooling still outperforms simple pooling, and both substantially outperform revenue optimized ranking in terms of relevance. We also compare to the same top-of-pool baseline described for Figure 1.

Enhanced Pooling:	Rel.	Irrel.	Uncov.	Total
Log-Based Subst.				
Relevant	26.2%	11.6%	0.2%	38.0%
Irrelevant	3.4%	3.4%	0.0%	6.8%
Uncovered	24.0%	29.0%	2.2%	55.2%
Total	53.6%	44.0%	2.4%	100%

Table 3: Comparison of the relevance of substitutions from log-based substitution and enhanced pooling.

Note that log-based substitution does not exhibit the unusual drop in performance at low coverage seen earlier. This suggests that the query substitutions generated with very high confidence are very relevant to the original query yet poor for selecting ads.

Additionally, by comparing to Figure 2, we see that all methods are better at finding relevant query substitutions than they are at selecting ads using query substitutions. It is likely that this is a limitation of query substitution in general, as selecting a single substitution for a user query limits the potential ads that can be shown. However, it could be addressed if we allow multiple substitutions per input query. For example, if we were to select a small number of substitutions per input query, ads could be served for any of them, increasing the supply of potential ads while remaining computationally efficient.

4.4 Combining Coverage

One of the limitations that we have seen repeatedly for log-based substitution is that it is unable to obtain substitutions for many of the queries. We now address the question of how to combine that method with ours to generate more relevant substitutions for a larger fraction of the queries. Table 3 shows the fraction of queries where each method either selects relevant substitutions, selects non relevant substitutions and does not generate any substitutions on our evaluation set of 500 queries.

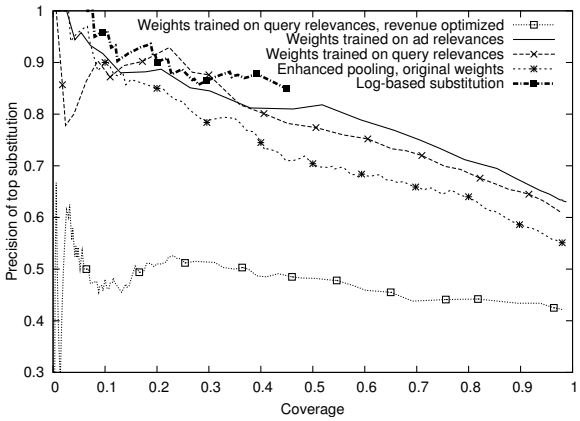

Figure 4: Relevance of query substitutions using weights retrained on the large judged dataset.

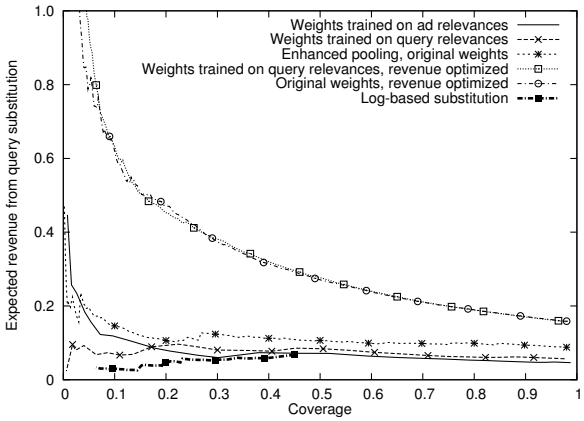

Figure 5: Expected revenue using weights retrained on the large judged dataset.

We see that enhanced pooling performs better on the queries that log based substitution is also able to generate substitutions on. However, for queries where log-based pooling is not applicable, enhanced pooling can still generate substitutions with reasonable performance. In fact, just as average substitution and ad quality can be improved by reducing coverage of queries, thresholding can be used to reduce the number of poor substitutions returned on queries uncovered by log-based substitution. For instance, enhanced pooling achieves a precision of 64% if only used to generate a substitution for 42% of the queries uncovered by log-based substitution. We plan to study such a combined substitution system in future work.

4.5 Tuning the Feature Weights

As described in Section 3, the feature weights used to rank candidate substitutions were trained using a small set of about 100 (query, substitution) pairs. We now observe that the substantially larger dataset collected for the 500 evaluation queries can also be used to improve the weights used to rank candidate substitutions. The feature weights can be trained either using the (query, substitution) judgments or using the (query, ad) judgments. We will now compare these two alternative training methods to determine which results in better performance.

We start by optimizing the weights to the (query, substitution) relevance judgments. Since the relevance of a substitution should be unrelated to the ads on the substitution, we trained a model omitting the features that depend explicitly on ads, again using a regression SVM with default settings. We then ranked candidate substitutions both with enhanced pooling, and revenue-optimized enhanced pooling.

To train the weights using the (query, ad) judgments, we set the target value of each (query, substitution) to the mean relevance judgment of the top three ads retrieved for that substitution. We also learned weights for two new features which were omitted in earlier experiments due to an oversight: *numberOfAds(t)* capturing the number of ads bidding on t, and *numberOfClients(t)*, the number of distinct advertisers who bid on t.

To evaluate the performance of these new ranking functions without overfitting, we performed the entire training and evaluation using 5-fold cross validation. Figure 4 shows the fraction of top substitutions that were found relevant to the original query. Note that as we had a fixed evaluation corpus, about one third of the substitutions now returned by the different methods had not been judged by the human judges, and thus were ignored.

We see that the relevance of substitutions selected by enhanced pooling, trained on either set of judgments, both when revenue optimized and not, has markedly improved over that seen in Figure 3. In particular, enhanced pooling now returns substitutions whose relevance almost matches those returned by log-based substitution. Computing the 95% binomial confidence intervals for the performance of the top 3 methods, we find that the difference in relevance between them is not statistically significant at any coverage level. In addition, the relevance of the revenue-optimized substitutions has also improved by about a third. Interestingly, despite this evaluation being in terms of the relevance of substitutions, optimizing to (query, substitution) judgments or the (query, ad) judgments makes little difference.

Evaluating the estimated revenue using the retrained weights, Figure 5 can be compared to the results obtained for the original weights and presented in Figure 1. We see that the performance of the revenue optimized method is essentially unchanged. However, the relevance for the simple pooling and enhanced pooling methods increased at the cost of expected revenue. Although the revenue for these methods has dropped, the estimated revenue using the weights trained on query relevances is still 20% to 30% higher than that generated by log-based substitution. These results reemphasize the earlier observation that revenue and ad relevance can be at odds.

We finish by studying the weights learned for the features using the evaluation data. We retrained the models evaluated in this section using all the relevance judgments, optimizing to the (query, substitution) judgments and the (query, ad) judgments. The weights learned are listed in Table 4.

Notice that the learned substitution functions place the most weight on the semantic and lexical features. We suspect that negative weights in many of the features are due partly to the correlation between some of the features and partly to random noise. For example, *shareWords* and *trigramCosine* are strongly related. In all, the net effect is that the largest contribution is due to simple textual similarity. However, the extent to which substitutions are representative of all candidate substitutions (represented by the *clientCosine* feature) is also important. Particularly interestingly, *maxMatchScore* has almost zero weight in both cases, suggesting that the original scores used to create the pool of

Feature	Target output	
	(query,sub)	(query,ad)
shareWords	-0.13	0.02
wordDistance	-0.01	-0.04
cosine	-0.04	-0.21
editDistance	0.00	-0.02
trigramCosine	0.28	0.14
maxMatchScore	-0.01	0.00
abstractCosine	0.46	0.38
taxonomySim	0.05	-0.02
queryFrequency	0.02	0.05
clientCosine	0.08	0.09
numberOfAds	–	-0.03
numClients	–	-0.03
maxBid	–	-0.07
secondBid	–	0.05

Table 4: Learned feature weights, optimized to the (query, substitution) and to the (query, ad) judgments.

candidate substitutions are not very useful given all the other features. Indeed, it indicates that the features just based on the bid phrases are more informative than those based on entire ads. Also, it is interesting that the revenue features have mostly negative weights. This means that the bid amounts and query frequency are slightly negatively correlated with substitution relevance.

5. CONCLUSIONS

We have presented a two-phase methodology for selecting advertisements to appear alongside Web search results. In an offline preprocessing phase we use several sources of external knowledge to build a query substitution table. Then, at matching time, retrieving suitable ads is performed nearly instantaneously by finding ads whose bid phrase exactly matches the substituted query. In contrast to previous query substitution studies, we optimize both the relevance of ads and the advertising revenue collected by the search engine. Our approach combines the flexibility of broad match with the computational efficiency of exact match. We have demonstrated that the expected revenue generated by our query substitutions is substantially higher than that offered by previous approaches, while the relevance of ads produced can be on par with the previous state of the art.

We plan to extend this work by studying in more detail the tradeoff between revenue and ad relevance. In particular, we plan to perform a real-world study showing advertisements generated by different techniques to Web search users. Furthermore, we plan to compare our method directly to revenue-optimized log-based substitution.

Finally, observe that the manual association of bid phrases to ads encodes a considerable amount of human knowledge, which we expect to be useful outside of the advertising domain. Consequently, in future work we intend to explore the effectiveness of bid phrases for general query expansion, cross-language information retrieval, and related tasks.

Acknowledgments

We would like to thank Rosie Jones, Benjamin Rey and Shaji Sebastian for providing access to the log-based query substitution implementation we used, as well as for helpful discussions about evaluation. We also thank the anonymous reviewers for useful feedback.

6. REFERENCES

[1] P. Anick. Using terminological feedback for web search refinement – a log-based study. In *SIGIR*, 2003.

[2] A. Broder, M. Fontoura, E. Gabrilovich, A. Joshi, V. Josifovski, and T. Zhang. Robust classification of rare queries using web knowledge. In *SIGIR*, 2007.

[3] A. Broder, M. Fontoura, V. Josifovski, and L. Riedel. A semantic approach to contextual advertising. In *SIGIR*, pages 559–566, 2007.

[4] A. Z. Broder, P. Ciccolo, M. Fontoura, E. Gabrilovich, V. Josifovski, and L. Riedel. Search advertising using web relevance feedback. *In preparation.*

[5] P. Chatterjee, D. L. Hoffman, and T. P. Novak. Modeling the clickstream: Implications for web-based advertising efforts. *Marketing Science*, 22(4):520–541, 2003.

[6] S. Cucerzan and E. Brill. Spelling correction as an iterative process that exploits the collective knowledge of web users. In *EMNLP*, pages 293–300, 2004.

[7] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990.

[8] 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*, 9(1):242–259, 2007.

[9] E. Gabrilovich and S. Markovitch. Feature generation for text categorization using world knowledge. In *IJCAI*, pages 1048–1053, 2005.

[10] T. Joachims. Making large-scale SVM learning practical. In B. Schlkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods*. MIT Press, 1999.

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

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

[13] P. Kowalczyk, I. Zukerman, and M. Niemann. Analyzing the effect of query class on document retrieval performance. In *Australian Conf. on AI*, pages 550–561, 2004.

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

[15] C. D. Manning and H. Schuetze. *Foundations of Statistical Natural Language Processing*. MIT Press, 2000.

[16] M. Mitra, A. Singhal, and C. Buckley. Improving automatic query expansion. In *SIGIR*, pages 206–214, 1998.

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

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

[19] E. M. Voorhees. Query expansion using lexical-semantic relations. In *SIGIR*, pages 61–69, 1994.

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

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

[22] C. Zhai and J. D. Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *CIKM*, pages 403–410, 2001.