The Sum of Its Parts: Reducing Sparsity in Click Estimation with Query Segments

Dustin Hillard, Eren Manavoglu, Hema Raghavan,
Chris Leggetter, Erick Cantú-Paz, and Rukmini Iyer

Yahoo! Inc,
701 First Avenue, Sunnyvale, CA
{dhillard,erenm,raghavan,cjl,erick,riyer}@yahoo-inc.com

Abstract. The critical task of predicting clicks on search advertisements is typically addressed by learning from historical click data. When enough history is observed for a given query-ad pair, future clicks can be accurately modeled. However, based on the empirical distribution of queries, sufficient historical information is unavailable for many query-ad pairs. The sparsity of data for new and rare queries makes it difficult to accurately estimate clicks for a significant portion of typical search engine traffic. In this paper we provide analysis to motivate modeling approaches that can reduce the sparsity of the large space of user search queries. We then propose methods to improve click and relevance models for sponsored search by mining click behavior for partial user queries. We aggregate click history for individual query words, as well as for phrases extracted with a CRF model. The new models show significant improvement in clicks and revenue compared to state-of-the-art baselines trained on several months of query logs. Results are reported on live traffic of a commercial search engine, in addition to results from offline evaluation.

Key words: query log mining, clicks, relevance, advertising

1 Introduction

In recent years search engines have increasingly provided an easy and accurate way for people to find sites and information on the web. A significant amount of the success of search engines can be attributed to their use of implicit data provided by users, such as links that form the web map, and more explicitly the clicks on search results in response to a user query. Leveraging this type of data has allowed search engines to provide nearly perfect results for many common searches, such as navigational searches that intend to find a specific website, or frequent informational searches as in looking for a popular movie or musician. The profitability of commercial search engines has also stemmed from the effective ranking of advertisements, which is heavily based on advertisements' historical click rates. A commercial search engine can easily rank advertisements for optimal revenue when sufficient historical data is available; the probability of a user clicking on an advertisement can be very accurately predicted when an ad has been presented for a particular query to millions, or even just hundreds, of search engine users.

The task of surfacing useful and accurate information becomes more challenging when there is less user feedback available. Ranking two similar websites is more difficult if neither is very connected to other known sites on the web. Similarly, choosing between two advertisements is more difficult if we have not seen how users respond to these ads in the past for a particular query. A small number of shorter search queries are very popular and common enough to collect robust statistics for, but the complete space of user queries is massive, as can be expected from the combinatorial explosion when queries contain long strings of words. In this work we quantify the quick decay in available historical data as queries become longer and more rare. This lack of sufficient history then motivates new approaches to modeling queries where little to no previous history exists. We propose a new approach that utilizes history for partial user queries, so that new or rare queries can benefit from history that has been observed for portions of the complete query. Our improved models show significant gains in prediction performance for rare queries, while also slightly improving common queries. In addition, we show that automatic segmentation of queries is much better than using just simple individual words. Our models focus on search advertising, where the information provided by clicks is particularly important.

We first present an overview of the sponsored search task in Section 2. Section 3 then describes our baseline models for predicting clicks and relevance. We motivate our new approach with analysis of historical click statistics in Section 4. Section 5 then describes our new approach for collecting history on partial search queries. We evaluate our new models offline in Section 6 and then in live user tests in Section 7. Previous work is discussed in Section 8. Finally, we summarize our results and conclude in Section 9.

2 Overview of Sponsored Search

The primary source of revenue for major search engines is from advertisements displayed during user searches. The online ad spend of advertisers has been growing significantly over the past few years [1]. A search engine typically displays sponsored listings on the top and the right hand side of the web-search results, in response to a user query. The revenue model for these listings is "pay-per-click" where the advertiser pays the search engine only if the advertisement is clicked. The advertiser "targets" specific keyword markets by bidding on search queries. For example, an advertiser selling "shoes" may bid on user queries such as "cheap shoes", "running shoes" and so on. Sponsored search offers a more targeted and less expensive way of marketing for most advertisers as compared to mass media like TV and newspapers, and has therefore gained momentum in recent years.

We now describe the search engine monetization (SEM) terminology used in this paper. The terminology is similar across most of the major search engines. An advertising **campaign** consists of many **ad groups**; each ad group in turn consists of a set of related keywords for a campaign. For each adgroup, there is a set of **bidterms** or keywords that the advertiser bids on, e.g., *sports shoes*, *stilettos*, *canvas shoes* etc. A **creative** is associated with an ad group and is composed of a **title**, a **description** and a **display URL**. The title is typically 2-3 words in length and the description has about 10-15 words. Clicking on an ad leads the user to the **landing page** of the advertiser.

An advertiser can choose to use **standard** or **advanced** match for the keywords in an ad group. For example, enabling only standard match for the keyword "sports shoes", will result in the corresponding creative being shown only for that exact query. Whereas, if the keyword is enabled for advanced match, the search engine can show the same ad for the related queries "running shoes" or "track shoes". A **bid** is associated with each keyword and a second price auction model determines how much the advertiser pays the search engine for the click [16].

Most search engines typically take a multi-stage approach to selecting and serving advertisements. The typical stages in a search advertisement system are as follows: (1) **retrieval:** finding a candidate set of ads for a query, (2) **relevance filtering:** a more complex second pass model that filters non-relevant ads for the query, and (3) **click through rate prediction:** estimating click through rate for the retrieved ads and ranking ads on the search page. A high level overview of our advertisement retrieval and relevance/click prediction system is shown in Figure 1.

2.1 Advertisement Retrieval

Past work on finding relevant ads for a query has typically used one of two different approaches: (a) a query rewriting approach [21, 27] or a (b) direct query-ad approach [10, 28]. In query rewriting, the goal is to generate a relevant rewrite q_j for a given query q_i. Then ads associated with the bidterm q_j are retrieved

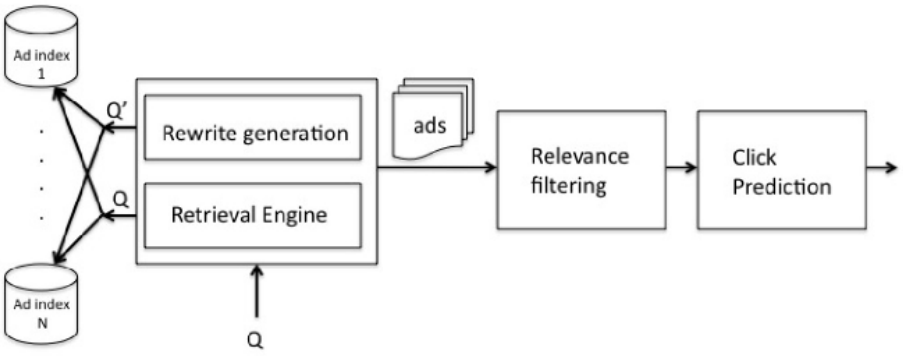

Fig. 1. A high level overview of a sponsored search system. Candidate ads are retrieved using various information retrieval techniques and then more complex models may be used for relevance filtering. Finally the click prediction system outputs the probability that an ad is likely to be clicked for a query. The output of the click model along with the advertiser bid is then used to determine the ranking and placement of ads on a search results page. The click and relevance prediction models make use of several features derived from user click feedback. The tables containing click through rate statistics are regularly updated.

in response to input query q_i. In direct query-ad matching the ads are treated as documents with multiple sections that correspond to the creative and perhaps the landing page. Following standard information retrieval and web-search methods, an input query is used to retrieve and rank candidate ads. We describe these different approaches briefly in this section to give the reader a picture of our overall system, but the remainder of our work will focus on relevance and click rate prediction.

Typical query re-writing approaches learn from user query transformations extracted from web-search logs [21, 39]. These transformations include similar queries and sub-phrases in query reformulations that are obtained from user sessions in the logs. These methods have been shown to work well in practice. One query re-writing approach used in our system that accounts for significant recall is a **LBQS (log based query substitution)** approach [21]. Taking an example from the original paper, given a query *"catholic baby names"*, the method considers rewrites of the original query such as *"baby names"* and *"catholic names"*. In addition rewrites are also generated by segmenting the original query as *(catholic) (baby names)* and considering insertions, deletions and substitutions of the individual segments in the query log giving alternative rewrites like *"(christian) (baby names)", "(baby names)"* etc. A log-likelihood ratio threshold is used to filter out query pairs that co-occur frequently by chance (due to the high query frequency of the individual queries). A second query rewriting approach used in our system is the collaborative filtering approach that learns rewrites from the click-graph. This system is described in detail in [5]. The differ-

ent query rewriting approaches are combined by a machine learning model which for a given query ranks the list of rewrites. The model computes the similarity of the new query to the original query taking into account lexical constraints and the frequency of the proposed transform in the logs. Rewrites above a certain score threshold are used to retrieve new ads.

In addition to query rewriting approaches described above, we also use a traditional information retrieval system, where ad creatives are indexed like documents. The retrieval model uses a query likelihood language model (LM) akin to the work of Ponte and Croft [25]. The system ranks documents by the probability that a document is relevant given a query. In our LM formulation the probability that an ad (a_i) i.e., a creative is relevant to a query (q) is given as [29]:

$$p(a_i|q) = \frac{p(q|a_i)p(a_i)}{p(q)} \tag{1}$$

In a multinomial approach, the words (w_i) in q are assumed to be i.i.d. Therefore, $p(q|a_i) = \prod_j^{|q|} p(w_j|a_i)$, where $p(w_j|a_i)$ is a smoothed probability of a word w_j being generated by the document a. All ads are stored in an inverted index and the entire corpus is scored in an effective and efficient manner at run-time. This language modeling framework is used in parallel with the query rewriting approaches to generate candidates to be sent to the relevance and click model.

While some of the query-rewriting techniques that are based on click-logs work well for queries commonly observed in the logs, the language modeling approach is effective even in the tail. On the other hand the language model based retrieval technique is restricted to word overlap between the query and the document whereas methods like the LBQS can retrieve a *sneakers* ad for a *running shoes* query. The different retrieval techniques are therefore complementary and in our experience we have found that pooling candidates from such diverse approaches works best for ad-retrieval. Ads obtained from the different retrieval techniques are de-duplicated and then sent to a relevance filtering and click-through rate prediction system which are described in the following sections.

2.2 Relevance Filtering

Our relevance filtering approach estimates a candidate ad's relevance in the context of a user query. The task is very similar to estimating relevance in web-search, but one major difference is that in sponsored search users have a lower tolerance for bad ads than they do for bad web-search results. We refer the reader to the study of Jansen and Resnick [19] for further details on user perceptions of sponsored search. Many queries do not have commercial intent. For example, displaying ads on a query like "formula for mutual information" may hurt user experience and occupy real-estate on the search results page in a spot where a more relevant web-search result might exist. Therefore, in sponsored search, we prefer not to show any ads when the estimated relevance of the ad is low. For this reason, in this paper we specifically focus on perceived relevance, meaning

the relevance of the ad as seen by the user, and not the final relevance of the ad's landing page. Beyond creating a bad user experience, irrelevant ads also create a poor advertiser experience by increasing costs to other advertisers. An irrelevant advertisement with a high bid can affect the cost paid by an advertiser whose ad is shown above this irrelevant ad in the second price auction. In this second stage we can apply a more complex relevance model that need not be constrained by the more limited set of features which we are restricted to in the optimized inverted index based ad retrieval during the first stage.

2.3 Click Through Rate Prediction

The final ad ranking displayed on the search engine is a product of the cost and the predicted Click Through Rate (CTR) of the ad. Given a set of ads $\{a_1...a_n\}$ shown at ranks $1...n$ for a query q on a search results page, the expected revenue is given as:

$$R = \sum_{i}^{n} P(click|q, a_i) \times cost(q', a_i, i) \tag{2}$$

where $cost(q', a_i, i)$ is the cost of a click for the ad a_i at position i for the bidterm q'. In the case of standard match $q = q'$, where for advanced match q' was selected in the retrieval phase. Most search engines rank the ads by the product of the estimated CTR ($P(click|q, a_i)$) and cost in an attempt to maximize revenue for the search engine. Therefore, accurately estimating the CTR for a query-ad pair is a very important problem. As discussed earlier, the primary data for predicting the probability of click are historical CTR statistics for query ad pairs that have been previously shown to users. However, there often may not be sufficient history because the ad inventory is continuously changing with advertisers adding, replacing and editing ads. Likewise, many queries and ads have few or zero past occurrences in the logs. These factors make the CTR estimation of rare and new queries a challenging problem.

3 Baseline Relevance and Click Models

The following sections describe the details of our state-of-the-art baseline models for ads in the context of a user query. We first describe our basic features and then two complimentary models, one that estimates relevance as judged by trained human editors, and one that estimates the probability of a user clicking on a given ad.

While relevance and clicks are highly related there are important differences. Editorial assessment of relevance typically captures how *related* an advertisement is to a search query, while click-through-rate (CTR) provides a signal about the *attractiveness* of an ad. The two measures can diverge: an ad to "Buy Coke Online" is highly related to the search "cocacola" although the CTR would likely be low because very few people are interested in buying Coke over the Internet; conversely an ad for "Coca Cola Company Job" is less related to the query, but could obtain a much higher CTR in our logs because the ad is highly attractive to users in a down economy. A more drastic example is an ad to "Lose weight now" that often receives a large number of clicks independent of what search term the ad is shown for (in most cases the ad would be judged to have low relevance to any particular search term). We focus on modeling click probabilities in order to estimate expected revenue and optimally rank candidate ads, but we also predict ad relevance in order to filter low quality ads.

3.1 Baseline Features

The two related modeling tasks share a common set of basic features, which can be categorized into two main groups: text features, and historical click rate features. Text features are designed to capture the impact of the perceived relevance of an ad compared to the user's search query. This group of features plays an important role for the query-ad pairs that occur rarely in our training data, but less so when sufficient historical features are available.

The basic text features incorporate 19 types: query length plus six features that each compare the query to the three zones of an ad (the title, description and display URL). These six features included word overlap (unigram and bigram), character overlap (unigram and bigram), unigram cosine similarity, and a feature that counted the number of bigrams in the query that had the order of the words preserved in the ad zone (ordered bigram overlap).

The second set of features, historical CTR features, are the most critical features used in our click models, and also provide a strong signal in relevance prediction. The past performance of a query-ad pair, when available, is a very accurate estimate of its future performance. The naive way of measuring historical CTR would be: $CTR(q, a) = clicks(\text{q,a})/imp(\text{q,a})$ where $imp(q, a)$ is the number of times where *query* and *ad* were shown together, and $clicks(q, a)$ is the number of times those impressions were clicked. However this definition would ignore the position-bias [20, 31, 14] completely. To account for the fact that users tend to examine certain parts of the page more than others, we use a position

normalized CTR metric known as clicks over expected clicks (COEC) [38]:

$$COEC(q,a) = \frac{\sum_p click_p(q,a)}{\sum_p imp_p(q,a) * CTR_p}, \quad (3)$$

where the numerator is the total number of clicks received by a query-ad pair; and the denominator can be interpreted as the expected clicks (ECs) that an average ad would receive after being impressed i_p times at position p. CTR_p is the average CTR for position p computed over all queries and ads. Note that the EC in this formulation does not only depend on rank, but it also depends on the section of the page the ad was shown in. While it has been shown that the COEC model is not perfect [11], it provides a significant improvement at almost no computational cost.

We collect COEC and EC statistics for every unique query and ad pair we observe. However, due to the dynamic nature of the domain (where advertisers are constantly changing their ads and campaigns to optimize their returns, and users are always searching for something new), we cannot have enough history to get reliable measures for most of the items we see in live traffic. To mitigate this problem, we measure clicks and expected clicks at different levels of specificity in query and ad space. As described in Section 2, advertisers organize their ads in hierarchies. An obvious way to increase the coverage of historical features would be to calculate clicks and expected clicks at various stages of this hierarchy, such as bidterm, ad group, campaign, domain, and advertiser. We also collect COEC and EC of an ad and the matching method used in retrieving it (standard or advanced), across all queries. Similarly we can collect only query dependent statistics. Such features can tell us something about the commercialness of the query. Finally, we also collect statistics for combined query and ad pairs, such as query-domain, query-adgroup, query-bidterm, and query-bidterm-creative.

We collect these COEC and EC statistics in the backend and update the online lookup tables every few hours. The backend process uses up to three months of history. We retire data that is older than three months due to privacy concerns. We also limit the number of observations used in the COEC and EC computation because we want to adapt to the changes in users' click behaviors quickly. For a frequent query ad pair, we might reach this limit in one day, and hence compute the statistics only for the last day, whereas we might need all three months of data for a rare item. This maximum observations threshold was tuned to optimize the offline model performance.

Text features and historical click rate features are used in both relevance and click modeling. However, there are features that make sense only in the context of click prediction. As discussed throughout this paper and in [20, 31, 14, 34], the same ad shown in different positions will receive different click rates, whereas its relevance will obviously not be changed. Presentation features are included to capture such biases in click modeling. We use the section the ad was shown in (i.e., above, below, or next to the organic results) and its rank within that section. We also added the index of the first occurrence of a query term in the

ad title and description as a feature, to capture the impact of changes in text presentation (such as bolding of query terms when such a match occurs).

Time of day and day of week are another set features that are used only in the click model. While the relevance of an ad to a query might also change over time, it usually is stable within such short periods, and the changes are not as periodic as the user's click rates.

3.2 Relevance Prediction Model

Based on the features described in the previous section, we learn a model of ad relevance. We can then use predicted relevance to improve our sponsored search system by filtering low quality ads (and also by providing a predicted relevance feature to the click model). Our relevance model is a binary classifier trained to detect relevant and irrelevant advertisements, given a particular search term. We have experimented with multiple learning approaches, but found Gradient Boosting Decision Trees (GBDT, [40]) to consistently perform the best.

The target for our models was generated from editorial data on a five point editorial scale (Perfect, Excellent, Good, Fair, Bad), where we consider all judgments better than "Bad" as relevant and the remaining "Bad" judgments as irrelevant ads. Judgments are performed by professional editors that achieve reasonable consistency. Our training set contains about 100k editorially judged query ad pairs. Our precision and recall results for detecting relevant ads are reported on an editorial test set of 50k query ad pairs. Training and test data were retrieved from our advertiser database with an ad retrieval system as described in Section 2.1 (an average of 10 ads is retrieved per query). The data contains 15k unique queries, which were selected based on a stratified sample of search engine traffic that represents all ten search frequency deciles.

Incorporating User Clicks in Relevance Modeling Our relevance model can predict relevance with reasonable accuracy based on just the simple text overlap features, but it will fail to detect relevant ads if no syntactic overlap is present. An ad with the title "Find the best jogging shoes" could be very relevant to a user search "running gear," but our baseline model has no knowledge that running and jogging are highly related concepts. We can introduce historical click rates for leveraging user click data to learn more semantic relationships between queries and ads.

Historical click observations for a query-ad pair can provide a strong indication of relevance and can be used as features in our relevance model. User click rates often correspond well with editorial ratings when a sufficient number of clicks and impressions have been observed. The relationship is however not deterministic (as discussed earlier), so we allow the model to learn how to incorporate observed click rates. Our baseline model incorporates the text and historical features as described in Section 3.1 above. Additional details and applications of the baseline relevance model can be found in [18].

3.3 Click Through Rate Prediction Model

We treat the click through rate prediction task as a supervised learning problem. Major search engines typically see tens of millions of queries a day, and a correspondingly large number of user clicks on ads as a response to these queries. We record the user response to each query-ad pair (1 if the user has clicked, 0 otherwise) that was presented to the user, and use this data to learn a model for predicting the CTR.

Various methods have been proposed in the recent years for the CTR prediction task. We can classify these methods into two broad categories: (1) session modeling approaches [20, 4, 15, 11, 17, 37, 34] where the focus has been on identifying the factors that affect users' behavior and using generative graphical models to explain those; (2) regression or binary classification methods where each query-document pair is an individual sample and all the factors that have an impact on the users' click behavior are captured in the features [31, 13, 33]. We chose the second approach because modifying the session models to handle previously unseen documents or queries by using features representations instead of the id's of the documents and queries would increase their complexity significantly. We chose a Maximum Entropy (maxent) model to estimate the CTR's because the learning approach can be efficiently parallelized, allowing us to easily scale the training to billions of examples.

The maxent model, also known as logistic regression, has the following form:

$$p(c|q, a_i) = \frac{1}{1 + \exp(\sum_{j=1}^{d} w_j f_j(q, a_i))}, \tag{4}$$

where $f_j(q, a_i)$ is the j-th feature derived for query-ad pair shown in position i (q, a_i) and $w_j \in \mathbf{w}$ is the associated weight.

Given the training set $\mathcal{D}$, we learn the weight vector $\mathbf{w}$ by maximizing the regularized likelihood:

$$\mathbf{w} = \max(\sum_{j=1}^{n} \log(p(c_j|q, a_i)) + \log(p(\mathbf{w}))), \tag{5}$$

where the first part represents the likelihood function and the second part uses a Gaussian prior on the weight vector $\mathbf{w}$ to smooth the maxent model [12]. Maximum likelihood estimation for maxent is a well studied problem [23]. In this work, given the large collection of samples and high dimensional feature space, we use a nonlinear conjugate gradient algorithm [24].

The maxent framework has many advantages, however learning the non-monotonic relationships between the features and the target or the non-linear relationships among features requires more engineering on the features side. In order to enable the model to learn non-monotonic feature-target relationships we quantize the continuous features. We use a simple K-means clustering algorithm with the following objective function:

$$\arg\min \sum_{j=1}^{k} \sum_{v_i \in C_j} ||v_i - u_j||^2, \tag{6}$$

where v_i is the feature value, u_j is the centroid of the cluster C_j and k is the number of clusters. We introduce binary indicator features for each cluster, and use a 1-in-k encoding as input in the maxent model. We also introduce a binary indicator feature to indicate that a certain value is missing, which is common for historical COEC and EC features. We used a grid search algorithm to identify the optimal number of clusters and found that the performance of the model was fairly stable for $10 \geq k \geq 20$. We compared the performance of the K-means based quantization to a decision tree based approach and did not see any significant difference in the final click prediction performance.

To model the relationships between different features, we create conjunction features by taking the Cartesian product of the binary indicators for pairs of features. Some of the conjoined features are selected using domain knowledge. We use conjunctions between COECs and expected clicks, for instance, because we expect that the COECs will be more reliable when ECs are high. Additionally we use an automated method to select from the list of all possible cross products. This iterative method looks at the improvement in log likelihood of the model on a held out data set for each candidate cross product feature, selects those candidates that provide the biggest gains, and starts the process again with the new features included to the baseline model. We also use a frequency cut off to eliminate the rare features. The threshold value is tuned to keep the minimum number of features without any significant degradation in the offline model performance.

Our baseline model contained approximately 20K features after quantization, conjunctions and selection. The model was trained on approximately 900M samples (sampled from 2 months of real traffic, which contains repeated query-ad pairs).

4 Sparsity in Click Logs

Our models for predicting relevance and clicks provide robust estimates when sufficient click history is available because most query-ad pairs do not change in relevance or click attractiveness very quickly. The challenging modeling task is to approximate these measures when we have not observed the event with sufficient frequency to make the simple prediction based on observed history. The frequency distribution for query-ad pairs has a long and heavy tail for low frequency occurrences, given the extremely large number of possible queries as compared to the number of queries that we actually observe in our click logs. In addition, we have a large number of potential advertisements (in the hundreds of millions) that can be paired with this large query space. The combination of these environments results in observed click logs that only contain few to no examples for most unique query-ad pairs.

This sparseness in observed events can be reduced by sharing information across entities to simplify the feature space. In the case of advertisers we can accomplish this by sharing click history for advertiser-created ad groups and campaigns, or for entire advertisers. In practice this provides some form of historical observation for most advertisements that we observe in our live system. We determine how often sufficient historical observations are available by analyzing a large sample of candidate ads for a query set that spans the range of query frequencies. For the purposes of this analysis, we use a threshold of three expected clicks to mean we have sufficient observations for a reliable estimate, although the trends are similar for any particular threshold. When we have seen three expected clicks in the logs we can assume some amount of useful information has been conveyed, and this also means we have typically observed 10s to 100s of impressions of an ad, depending on the rank it was displayed at. At the advertiser account level we observe 98% historical coverage, 97% at the display URL level, 69% at the ad group level, 63% at the creative level, and 29% at the individual bidterm-creative ad level. We do not directly address the issue of sparseness in the advertiser space for the remainder of this paper because we have reasonable coverage, although there is room for better approaches to sharing history across advertisements.

The spareness of historical observations in the query space is a much larger issue, primarily due to the large set of possible queries. When history is aggregated across queries we find that 61% of our representative set has greater than three expected clicks, which is significantly less when compared to the broad advertiser history levels. Figure 2 illustrates the impact of query history on the performance of the baseline click model. We see that the accuracy of the model drops drastically when less historical query observations are available.

The most specific and accurate historical observations are for individual query-ad pairs, but coverage is minimal with sufficient observations for only 3%. As discussed in Section 3.1, we can aggregate the query with the various levels of the advertiser hierarchy to obtain additional coverage. We find 6% coverage at the query-ad group level, 8% at the query-display URL level, and 11% coverage at the query-bidterm level. This low coverage of historical observations

for most query-ad level aggregates motivates approaches to better share observed history across queries. The next section proposes a new method for aggregating historical features based on partial queries, which can allow for much broader coverage of the user query space.

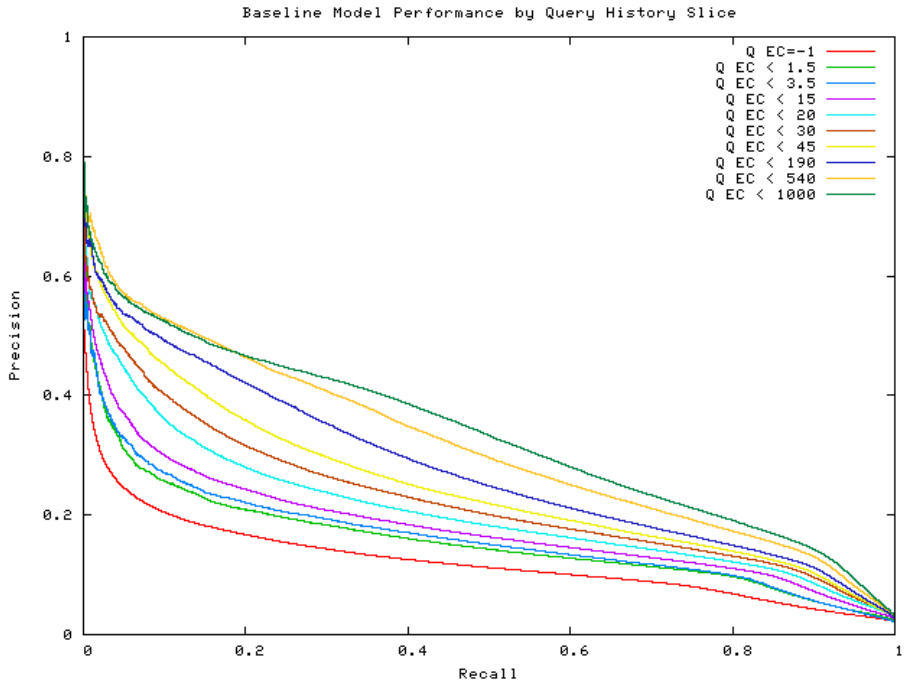

Fig. 2. Change in Precision-Recall for varying levels of Query History.

5 Click History for Partial Queries

To address the specific problem of overcoming the sparsity of click data for rare queries, we break a query into individual segments or phrases ($s_1..s_n$) and estimate COEC using the weighted average of historical segment level COEC information for the query segments s_i and ad, a, pairs as follows:

$$\widehat{COEC}(q,a) = \sum_i COEC(s_i,a)P(s_i|query) \tag{7}$$

$COEC(s_i,a)$ is similar to Equation 3, except that we aggregate clicks and impressions per segment s_i and ad a pair, rather than per query and ad pair. When expected clicks ($EC = \sum_p imp_p(s_i,a) * CTR_p$) for a particular segment s_i and ad a is 0, we can then back off to the average value for that segment, i.e. $EC(s_i)$.

5.1 Query Segmentation

In order to detect segments s_i and estimate the probability of the segments being generated from the query ($P(s_i|query)$), we first use a probabilistic segmenter to estimate a distribution $P(segHyp_j|query)$ over all possible segmentation hypotheses $segHyp_1...segHyp_m$ of a query (see Table 1 for an example).

Given a probability distribution over possible segmentation hypotheses of a query, we get:

$$P(s_i|query) = \sum_j P(s_i|segHyp_j)P(segHyp_j|query) \tag{8}$$

We now describe the procedure for estimating $P(segHyp_j|query)$ and $P(s_i|segHyp_j)$ respectively.

Estimating $P(segHyp_j|query)$: Several models for probabilistic segmentation of sequences of text exist in the literature (eg., [26, 22]). These sequential models typically aim to label the tokens $q_1, q_2...q_n$ (of a piece of text such as a query/document) with a sequence of labels y_1, y_2 $...y_n$. For our task y_i is a binary indicator variable where a value of 1 denotes the beginning of a segment (B-SEG) and 0 denotes that the token is the continuation of a segment (I-SEG). Let $segHyp_j = y_{j1}...y_{jn}$ denote a possible segmentation hypothesis of the query (see Table 1 for an example).

Among the many probabilistic segmentation models that exist in the literature, Conditional Random Fields (CRFs) have typically been quite successful [22]. The conditional probability $P(segHyp_j|query)$ is given as:

$$p(segHyp_j|query;\Lambda) = \frac{1}{Z(query;\Lambda)} exp\{\sum_k \lambda_k \sum_i^n f_k(y_{i-1}, y_i, query, i)\} \tag{9}$$

segmentation hypothesis	$p(segHyp_j\|query)$
|new york times square|	0.552
|new york | times square|	0.394
|new york | times | square|	0.135

Table 1. The above example illustrates the top 3 segmentation hypotheses ($segHyp_j$) for an example query.

where $f_k(y_{i-1}, y_i, query, i)$ is a feature function and $\Lambda = \{\lambda_i\}$ are the learned feature weights. We have several features based on dictionaries of people names, celebrity names, stopword lists and the top phrases in the query logs. CRFs are particularly attractive because we can use arbitrary feature functions on the observations.

We use the CRF++ toolkit [1] to train the model. The model is trained in a supervised fashion on 6000 queries annotated by humans. The human annotators were given queries that were pre-segmented by an unsupervised Hidden Markov Model (HMM) similar to [35]. The annotators were then asked to correct the segmentations for errors. We have found that agreement rates among annotators are typically around 80% for this task. We evaluated our segmentation models with Word Error Rate (WER), which is the error rate of classifying an individual word into B-SEG or I-SEG classes. The HMM and the CRF both had 14% WER on a separate held out set of 3000 queries. The WER of a naive unigram classifier that always predicted B-SEG was 28%.

While WER is reasonably indicative of performance, it does not accurately represent the performance of the segmenter for phrases that are important for advertising. Therefore, we also obtained a different data set of about 1000 queries where annotators were asked to mark key words and phrases that are important for preserving the original meaning of the query and that must be matched in a relevant advertisement. For example, a query like "how do i find parking around san jose" may be annotated as "how do i find (parking) around (san jose)". This second evaluation data set allows us to measure the the accuracy of the different segmentation approaches in detecting the key phrases that are central to the meaning and intent of the query using a metric we call *phrase-WER*. To compute phrase-WER we go over each of the key phrases in the query and for each phrase measure the error rate of classifying its individual words as B-SEG or I-SEG. The WER of the phrases in a query are averaged to give an average WER per query. The final phrase-WER metric for the test set is the overall mean computed over each query's phrase-WER. Using this new metric, the HMM and CRF have error rates of 24% and 19% respectively (with 28% for the unigram model). If we compute this metric strictly on multiword phrases, the two methods have error rates of 19% and 18% respectively (with 45% for the unigram model). Given the better performance of the CRF model across

[1] `http://crfpp.sourceforge.net/`

multiple metrics and test sets we use it throughout the rest of our downstream experiments.

Alternatively, we also found that dictionary based segmentation can achieve accuracy similar to that of the CRF, but is highly dependent on the quality of the dictionary. The CRF on the other hand requires much less manual tuning and can generalize better to unseen phrases. In addition, machine learned models provide n-best segmentation hypotheses with probability values that can then be directly used by our downstream model.

Estimating $P(s_i|segHyp_j)$: We assume a maximum likelihood model to estimate the probability of a segment s_i being observed given a particular segmentation hypothesis ($segHyp_j$) of a query as:

$$\begin{aligned} P(s_i|segHyp_j) &= 1/|segHyp_j|; \forall s_i \in segHyp_j \\ &= 0 \text{ otherwise} \end{aligned}$$

where $|segHyp_j|$ is the number of segments in a segmentation hypothesis ($segHyp_j$). This assigns uniform probability to each segment within a segmentation hypothesis.

5.2 Aggregating Segment Level Historical CTR

We can now collect new click-rate statistics at the query segment level, as opposed to the baseline model that only collects statistics for whole queries. We weight query segments by their predicted segmentation confidence and proceed to collect click and expected click counts for various aggregations. In this work we collect statistics for all query segments, all query segment and ad pairs, as well as query segment and ad domain pairs. The online lookup tables contain on average 15 million unique query segments, 40 million unique query segment and domain pairs, and 70 million unique query segment and ad pairs. The n-best segmentations from the CRF model are used in Eq. 7 to estimate $P(click|query)$, $P(click|query, adid)$ and $P(click|query, domain)$.

We investigate the use of several other features derived from $COEC(s_i, a)$: maximum $COEC(s_i, a)$ per query, minimum $COEC(s_i, a)$ per query, the number of segments for which the ad id had historical CTR data, and the number of segments for which the domain had historical CTR data. One can view Eq. 7 as a weighted average of $COEC(s_i, a)$'s since $\sum_i P(s_i|query)$ adds up to 1. Given this perspective we also computed the weighted geometric mean of $COEC(s_i, a)$'s using $exp(\sum_i P(s_i|query)log(P(click|s_i, a))$.

In order to assess the utility of the segmentation algorithm we also collected click-rate statistics with a naive segmentation where each segment is a single word. Here we again used Eq. 7 to estimate the COEC based on unigram clicks and expected clicks. We also generated the same set of additional features described above using these unigram COEC aggregates.

6 Offline Evaluation

6.1 Relevance Prediction

We report performance for human labeled query-ad pairs judged on a standard five point relevance scale (Perfect, Excellent, Good, Fair, Bad). For training and evaluating our filtering classifiers we mark all instances with a rating of "Fair" or better as *relevant* and we measure precision and recall of relevant ads.

Figure 3 presents the precision and recall curves for three models: the baseline model with standard text and historical features (*baseline*), the baseline model with additional features based on segment overlap (*textCRF*), and the baseline model with both segment text overlap and segment click feedback features (*textCRF+clickCRF*). The *textCRF* features compute the same set of basic text features described in Section 3.1, but treat the predicted segments as atomic units, whereas the baseline features use simple words as atomic units. The *textCRF* model slightly outperforms the *baseline* model at the maximum F-Score point ($p < 0.02$). The combined *textCRF+clickCRF* significantly outperforms the baseline model ($p << 0.01$);

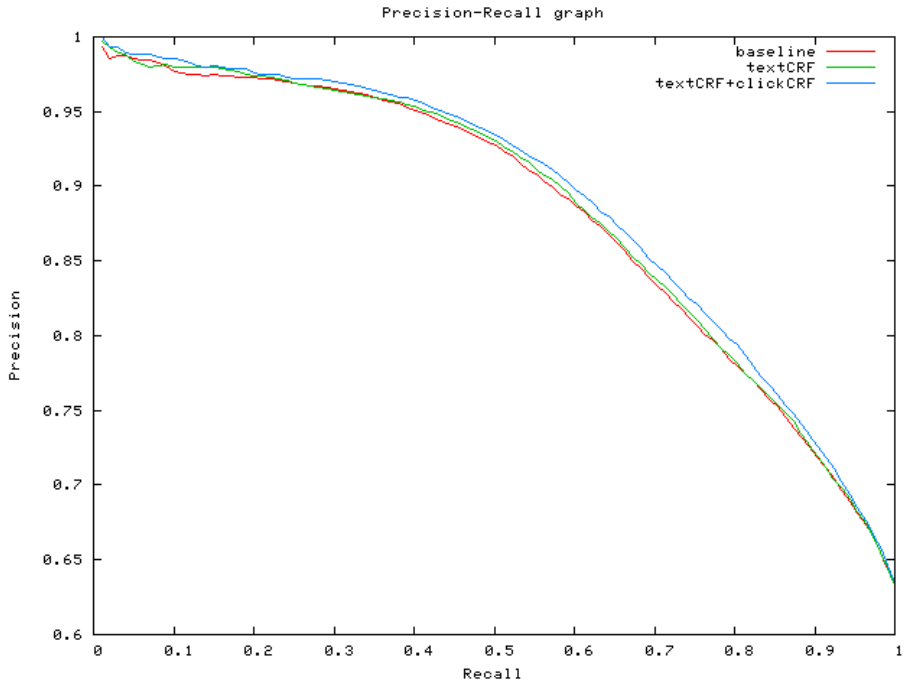

Fig. 3. Precision Recall with text-based and click-based CRF features

Table 2 provides the distribution and filtration rates for the five relevance grades for each model. For each model we pick the operating point that keeps the bad filtration rate constant and measure whether the new model retains more Perfects, Excellents, Goods and Fairs. The segment overlap features in *textCRF* show some improvement in recall, but the *clickCRF* features provide the largest improvement by decreasing the Excellent, Good and Fair filtration rates as compared to the baseline. Adding the two together (*all*) provides some recall for "Good," but segment click features provide the primary gains.

Grade (# instances)	baseline	textCRF	clickCRF	all
1) Perfect (200)	**0.016**	**0.016**	0.026	0.032
2) Excellent (250)	0.197	0.185	**0.168**	**0.168**
3) Good (5k)	0.154	0.150	0.139	**0.128**
4) Fair (18k)	0.308	0.304	**0.289**	0.292
5) Bad (18k)	**0.680**	**0.680**	**0.680**	**0.680**

Table 2. Filtering rates (numbers in parentheses indicate the approximate number of query-ad pairs). Lower values for Perfect, Excellent, Good and Fair are better and a higher filtration rate for Bad is better.

6.2 Click Prediction

We trained a new click model with the segment based click-rate features described in Section 5.2 and compared it to the baseline model that was introduced in Section 3.3. We used the same training data set, same K-means parameters, and the same frequency threshold for both models. We tested the models on a data set which was sampled from two weeks of real user traffic (100M query-ad pairs). The sampling process was designed such that the users who appeared in the training data were excluded from the test data, and the two weeks that were used in testing proceeded the training period. We used log likelihood to measure how well the models fit the data, and area under the precision recall (P-R) curve to determine how effectively we can predict the clicks.

Table 3 presents the results of three models for various levels of query history. To slice the data by query history we use the Query EC (expected clicks) as a surrogate for frequency. The concept of EC was introduced as part of Equation 3 and denotes how often the query has been observed with ads in the logs. A value of -1 indicates that the query was not present in the online lookup tables at the time of the request. In Table 3 we show results for the unigram model that uses the unigram based click-rate features, the segment model that uses the segment based click-rate features and the combined model that was trained using both sets of features. In this table we show the performance of these models compared to the baseline model described in Section 3.3. We see that all three models show nice gains for the cases where Query was unknown (see Figure 4 for a closer view). The unigram model shows about 4.7% improvement in P-R (measured by area under the curve, AUC) and the segment features almost

Slices	Unigram Model		Segment Model		Combined Model	
	LL	PR	LL	PR	LL	PR
Q EC=-1	0.8%	4.7%	1.4%	8.5%	1.5%	**8.8**%
Q EC=1.5	0.7%	3.1%	1.2%	6.3%	1.2%	**6.4**%
Q EC=3.5	0.5%	2.6%	0.9%	4.4%	0.9%	**4.6**%
Q EC=12	0.4%	2%	0.6%	3.4%	0.7%	**3.6**%
Q EC=45	0.1%	0.2%	0.2%	0.6%	0.2%	**0.6**%
Q EC=540	-0.04%	-0.1%	0.04%	0.0%	0.05%	0.0%
Q EC=2328	-0.1%	-0.2%	0.02%	0.0%	0.02%	0.0%
Q EC=5000	0.0%	0.0%	0.1%	0.7%	0.1%	**0.8**%

Table 3. Improvement in log likelihood (LL) and AUC of the Precision Recall Curve (PR) for varying Query History Slices. Most of the gains in the combined model are obtained from the segmentation model.

Slices	Unigram Model		Segment Model		Combined Model	
	LL	PR	LL	PR	LL	PR
Standard Match	0.1%	0.1%	0.2%	0.4%	0.2%	0.5%
Advanced Match	0.5%	2.5%	1%	4.7%	1%	4.9%

Table 4. Improvement in log likelihood (LL) and AUC of the Precision Recall Curve (PR) by Matching Method.

double the gain (8.5%). Although combining the two sets of features show some additional gains, the difference compared to the segment model is fairly small. We see that the unigram model starts showing some degradation for queries with higher expected clicks (Q EC=540 and 2328), however the segment model does not share this pattern. In fact, if anything, we see gains with the segment model in these higher EC slices.

In Table 4 we show the performance of the models for standard and advanced match. The segment model, again, shows much larger gains compared to the unigram model, and the combined model performs very much like the segment model. Gains are much larger in the advanced match slice. This behavior is expected because advanced match is a more relaxed matching method and therefore covers less frequent query-ad pairs. The unigram model is neutral for the standard match, but we see some small yet statistically significant gains ($p < 0.01$) with the segment model.

7 Online Evaluation

We further tested the segment based relevance and click models on live traffic. Given the large user base of the commercial search engine that we had access to, it was possible to expose our algorithm to several million users and reliably collect statistics over a sufficiently long period of time. Using a large number

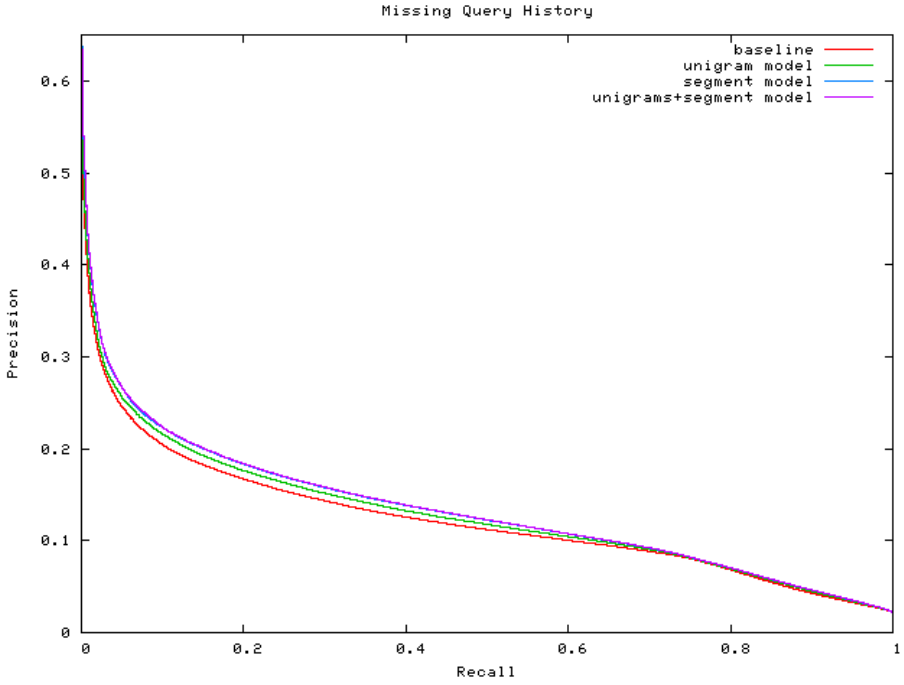

Fig. 4. Precision/Recall curve for No Query History Slice

of page-views containing candidates ranked and filtered by our proposed system and the baselines, we can reliably estimate measures that indicate the quality of the individual systems. In particular, this large-scale live testing provides the opportunity to evaluate on many new queries which have never been seen before in our historical logs. Additionally, because in "pay-per-click" advertising the desired goal of the search engine is user-clicks on ads, metrics from such "bucket-testing" can help evaluate the monetization capability of the new algorithm.

Our online experimentation environment is very similar to the single-layered design described in [36]. In a nutshell, in bucket tests the user population is divided into random subsets (or "buckets"). The users who are assigned to the control bucket are shown the ads from the baseline system, whereas the users in the experimental buckets are exposed to new retrieval, filtering or ranking algorithms. This allows us to compare the effectiveness of the new methods to the baseline (or production) system by directly measuring user metrics, such as clicks per search, as well as business metrics like revenue per search. The randomization algorithm used in assigning the users to buckets ensures that for two identical buckets the metrics will be indistinguishable.

Below we report the results of our online experiments. We ran the baseline, the new relevance model, and the segment click model with the new relevance model for several weeks on a sufficient portion of the live search traffic.

7.1 Relevance Prediction

Based on our offline analysis of the relevance model described in Section 6.1 we expect to see an improvement in recall with the *clickCRF* relevance model. To test this hypothesis we plot the precision recall curves of the baseline model and the *clickCRF* model using logged click data for each experiment. In Table 5 we present the change in the area under the precision-recall curve for various query history levels. We stopped at query EC = 10, because there was no significant change for queries with higher ECs. We see gains in all slices, except for the case where query EC is between 3 and 5. This set is the smallest group in our evaluation set and shows large variations in the lower recall range. The rest of the query EC slices show a common trend: large improvements in precision for the lower recall regions. Figure 5 shows this behavior for the query EC < 3 slice. If we wanted to get 50% precision with the baseline model we would have to lower the recall to 2.5%, however when filtering with the *clickCRF* relevance model we can get the same precision in click prediction at 7.5% recall.

7.2 Click Prediction

In Figure 6 we present the precision-recall performance of the baseline and the *segment* click model on the missing query history (Q EC=-1) slice. Comparing this to Figure 4, the offline analysis of the same slice, we can say with confidence that the online performance of the model matches what we observed in our offline setup.

Query EC Slice	% Change in P-R AUC
Q EC=-1	+2.6%
Q EC $\leq$ 1	+2.2%
Q EC $\leq$ 3	+15%
Q EC $\leq$ 5	-2%
Q EC $\leq$ 10	+2%

Table 5. Click Precision-Recall performance of the *clickCRF* Relevance Filter.

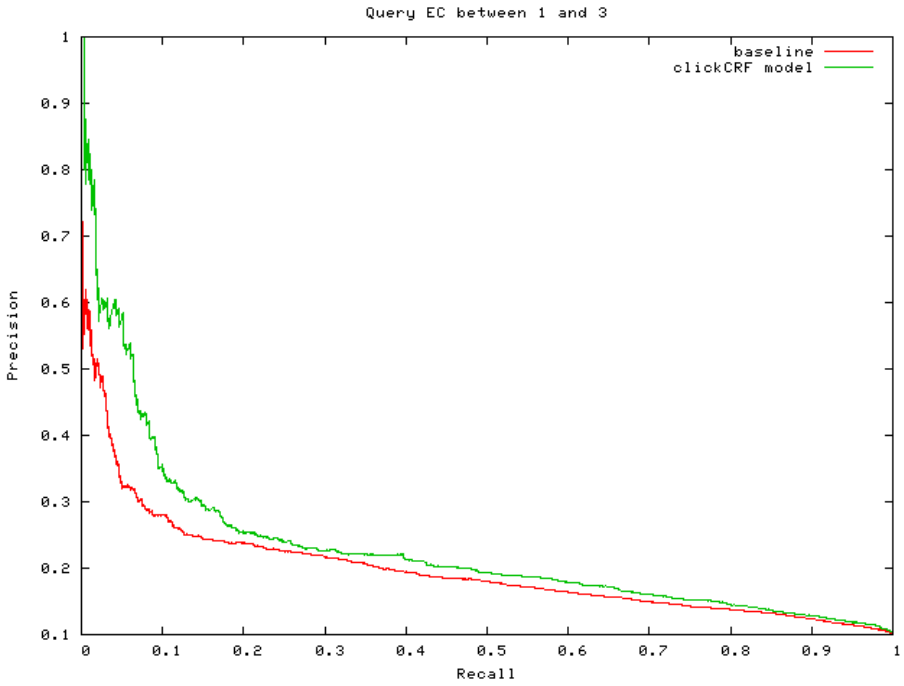

Fig. 5. Click Precision/Recall curve with *clickCRF* Relevance Filter for query EC < 3.

As mentioned in Section 2, the predicted CTR is also used in revenue estimation and pricing. Therefore the accuracy of the CTR predictions must also be measured. In Figure 7 we compare the predicted CTR (y axis) to the observed CTR (x axis) for the queries with no history. Each point in this correlation graph corresponds to a bin of query-ad pairs. The query-ad pairs were grouped into bins by using an equal size binning algorithm after being sorted by their predicted CTRs. The size of the bins was calculated using the normal approximation interval (with 95% confidence). We see that even though the new model still over predicts the CTR, the correlation of the new *segment* click model is improved (from 0.986 to 0.992, 0.6%). We see a 20% reduction in the mean squared error of the predicted CTR.

The gains of the *segment* click model are not limited to queries with no or little history. Figure 8 shows the correlation graph for queries with EC's between 10 and 100. We see 0.45% increase in the correlation and 40% decrease in the mean squared error rate, comparable to the gains we saw for queries with no history.

Overall, we found that the combined impact of the models provided an increase in clicks per search of 1.5% relative, and an overall revenue per search gain of 2% relative (from an additional slight increase of 0.5% in price-per-click). The increased recall of relevant and clickable ads, combined with the significant improvements in click model predictions, provides increased clicks and revenue from the sponsored search system. For large commercial search engines a 2% relative increase in revenue per search represents a substantial increase in revenue, and this improvement is built on top of a state-of-the-art baseline that is highly optimized.

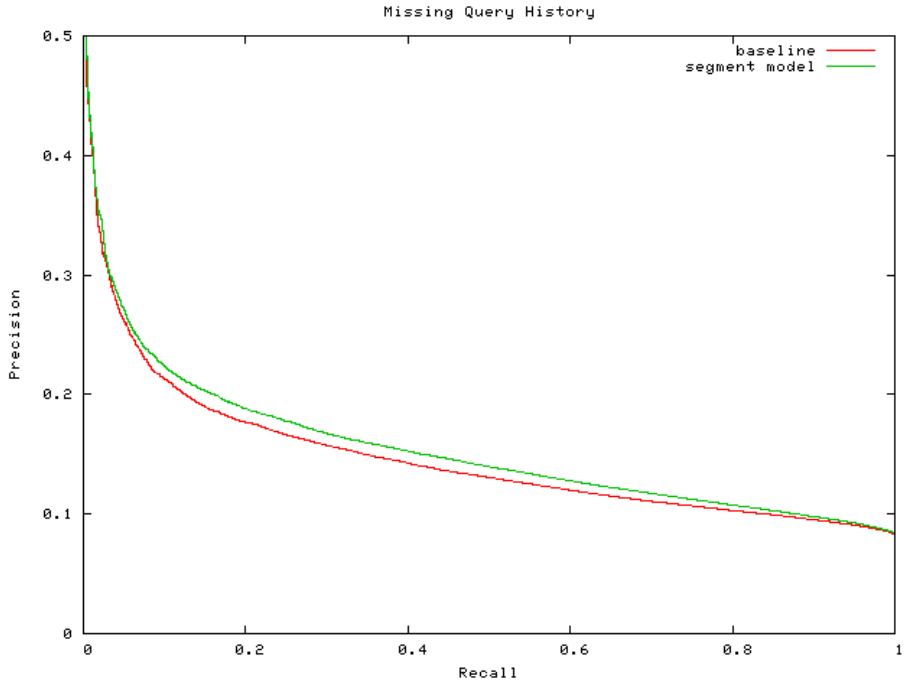

Fig. 6. Precision/Recall curve of the *Segment* click model and baseline for missing query history slice. The graph represents performance of the new approach on live traffic.

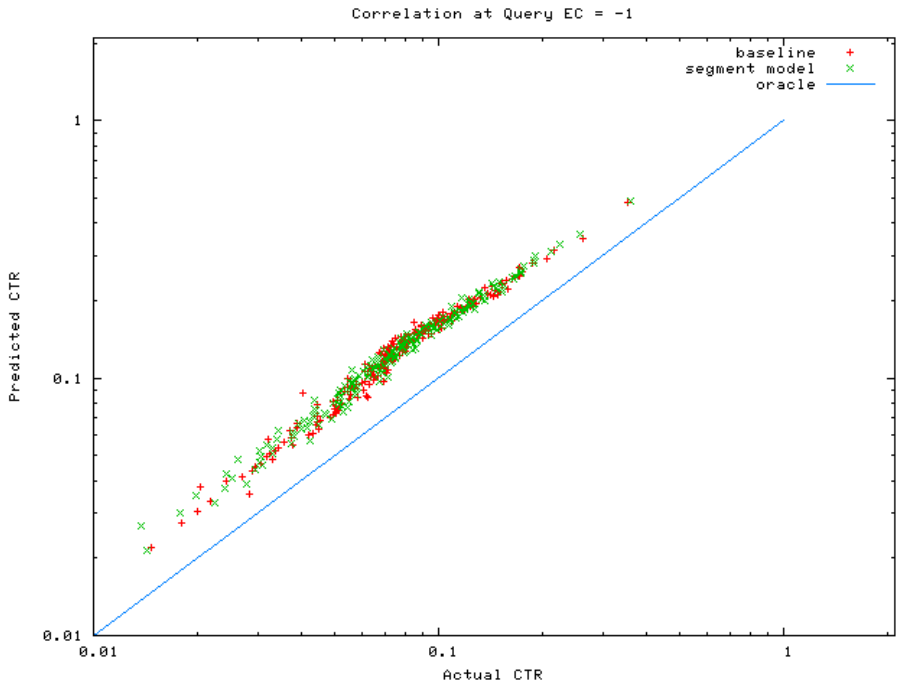

Fig. 7. Correlation graph of the *segment* click model for missing query history slice. The correlation of the new *segment* click model is improved (from 0.986 to 0.992, 0.6%).

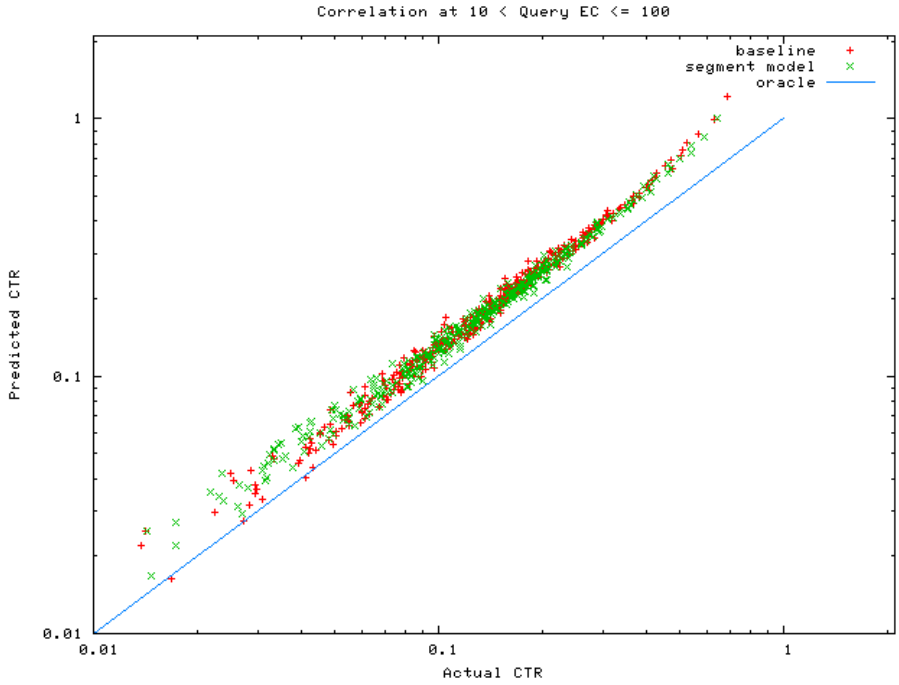

Fig. 8. Correlation graph of the *segment* click model for 10 < query EC <= 100.

8 Related Work

While much work has been done on click modeling for search and advertising in the recent past [20, 31, 13, 34], few have focused on estimating the click rates of rare events. In Agarwal et al. [3], the authors use an existing hierarchy to estimate CTRs for ads in the content match problem (the task of showing a relevant ad on a publisher page). Their work assumes a pre-existing classifier that can categorize a page and an ad into a taxonomy. In our domain the equivalent of a page is a query and given that a typical web-query is 2-3 words in length, it is difficult to accurately classify queries into a taxonomy. Recent work by Agarwal [2] has developed scalable log-linear models that couple parameters based on feature hierarchies, which helps improve learning for rare events. This recent work is complementary to the new features we develop in our approach, and it would be interesting to combine the approaches in future work.

Zhou et al. [41] addressed the issue of sparseness in the document retrieval domain. They utilized the CTRs of query substrings to estimate the probability of click at the query-document level. They used a greedy algorithm to combine the words in a query to form hierarchical ngrams and then estimated the parameters to combine ngram CTRs at every level of the hierarchy. The authors' estimation techniques did not take into consideration the position bias issue in the click logs. We would have liked to evaluate their method to combine segment level COECs but unfortunately a straightforward implementation of their method requires significantly more space and effort online than the method described in this paper, making it impractical for us to test it on live traffic.

Other studies have used attributes of a query to improve CTR estimation. For example, Ashkan et al [6] use query intent analysis to improve CTR estimation for sponsored search. The click prediction models of [32, 31] also report gains from features that characterize the query by categories or clusters, but they have not specifically described their approach or assessed the impact in the context of sparse historical click logs.

Fain and Regelson [30] predict the click-through rate for rare or novel terms by clustering bid terms in a hierarchy, but they were not looking at the same click and relevance prediction tasks as modern sponsored search systems. In their work the CTR of a term is predicted based on the CTRs of terms in the same cluster and the parent clusters. Work by Baeza-Yates [7] clustered queries based on clicked documents, and found improvements for query recommendation and result ranking. Their experiments use a relatively small data set of clicked queries (of about 30k queries), so the approach does not provide coverage of rare queries. We have experimented with various clustering approaches internally (such as [8]), but have not found any that provided significant improvements. In addition, assigning new and rare queries to clusters at run time is not straightforward and can be computationally expensive.

Rare queries were also addressed in the context of improving retrieval models for advertising [9]. However these works do not address the CTR estimation problem for such queries.

9 Conclusions

We presented an approach for improving sponsored search relevance and click prediction with features derived from CRF segmentation of user queries. We conducted offline and online experiments on large amounts of real user traffic and showed that the new features significantly improve the recall of the relevance filter and the prediction accuracy of the click model compared to state-of-the-art baselines. We also compared the CRF segmentation to a simplistic unigram model to verify our intuition that the gains coming from using these features exceed the gains possible from using simple word based click features.

The online experiments demonstrate that our proposed models provide significant gains in clicks and revenue, which results in considerable impact for commercial search engines. Our approach successfully reduces the sparsity of the complex space of user search queries, providing broad improvements to models of relevance and clicks.

References

1. www.emarketer.com/Article.aspx?id=1006319.
2. D. Agarwal, R. Agrawal, R. Khanna, and N. Kota. Estimating rates of rare events with multiple hierarchies through scalable log-linear models. In *KDD*, pages 213–222, 2010.
3. D. Agarwal, A. Z. Broder, D. Chakrabarti, D. Diklic, V. Josifovski, and M. Sayyadian. Estimating rates of rare events at multiple resolutions. In *KDD*, pages 16–25, New York, NY, USA, 2007. ACM.
4. E. Agichtein, E. Brill, and S. Dumais. Improving web search ranking by incorporating user behavior information. In *SIGIR*, 2006.
5. T. Anastasakos, D. Hillard, S. Kshetramade, and H. Raghavan. A collaborative filtering approach to ad recommendation using the query ad click graph. Technical Report YL-2009-006, Yahoo! Labs, Aug 2009.
6. A. Ashkan, C. L. A. Clarke, E. Agichtein, and Q. Guo. Estimating ad clickthrough rate through query intent analysis. In *WI-IAT '09: Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, pages 222–229, Washington, DC, USA, 2009. IEEE Computer Society.
7. R. Baeza-Yates, C. Hurtado, and M. Mendoza. Improving search engines by query clustering. *Journal of the American Society for Information Science and Technology*, 58(12):1793–1804, 2007.
8. D. Beeferman and A. Berger. Agglomerative clustering of a search engine query log. In *KDD*, 2000.
9. A. Broder, P. Ciccolo, E. Gabrilovich, V. Josifovski, D. Metzler, L. Riedel, and J. Yuan. Online expansion of rare queries for sponsored search. In *WWW*, pages 511–520, 2009.
10. A. Z. Broder, P. Ciccolo, M. Fontoura, E. Gabrilovich, V. Josifovski, and L. Riedel. Search advertising using web relevance feedback. In *CIKM*, 2008.
11. O. Chapelle and Y. Zhang. A dynamic bayesian network click model for web search ranking. *WWW*, 2009.
12. S. Chen and R. Rosenfeld. A gaussian prior for smoothing maximum entropy models. Technical report, Carnegie Mellon University, 1999.
13. M. Ciaramita, V. Murdock, and V. Plachouras. Online learning from click data for sponsored search. In *WWW*, 2008.
14. N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. *WSDM*, 2008.
15. G. E. Dupret and B. Piwowarski. A user browsing model to predict search engine click data from past observations. In *SIGIR*, 2008.
16. 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), March 2007.
17. F. Guo, C. Liu, A. Kannan, T. Minka, M. Taylor, Y. Wang, and C. Faloutsos. Click chain model in web search. *WWW*, 2009.
18. D. Hillard, S. Schroedl, E. Manavoglu, H. Raghavan, and C. Leggetter. Improving ad relevance in sponsored search. In *WSDM*, 2010.
19. B. Jansen and M. Resnick. Examining searcher perceptions of and interactions with sponsored results. In *Workshop on Sponsored Search Auctions*, 2005.
20. T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *SIGIR*, 2005.

21. R. Jones, B. Rey, O. Madani, and W. Greiner. Generating query substitutions. In *WWW*, 2006.
22. X. Li, Y.-Y. Wang, and A. Acero. Extracting structured information from user queries with semi-supervised conditional random fields. In *SIGIR*, pages 572–579, 2009.
23. T. Minka. A comparison of numerical optimizers for logistic regression. Technical report, Microsoft, 2003.
24. A. Mordecai. *Nonlinear Programming: Analysis and Methods.* Dover Publishing, 2003.
25. J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *SIGIR*, 1998.
26. L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, pages 257–286, 1989.
27. F. Radlinski, *A. Broder*, P. Ciccolo, E. Gabrilovich, V. Josifovski, and L. Riedel. Optimizing relevance and revenue in ad search:a query substitution approach. In *SIGIR*, 2008.
28. H. Raghavan and R. Iyer. Evaluating vector-space and probabilistic models for query to ad matching. In *SIGIR '08 Workshop on Information Retrieval in Advertising (IRA)*, 2008.
29. H. Raghavan and R. Iyer. Probabilistic first pass retrieval for search advertising: From theory to practice. In *CIKM*, 2010.
30. M. Regelson and D. C. Fain. Predicting click-through rate using keyword clusters. In *In Electronic Commerce (EC)*. ACM, 2007.
31. M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: estimating the click-through rate for new ads. In *WWW*, 2007.
32. D. Sculley, R. G. Malkin, S. Basu, and R. J. Bayardo. Predicting bounce rates in sponsored search advertisements. In *KDD*, pages 1325–1334, 2009.
33. B. Shaparenko, O. Cetin, and R. Iyer. Data driven text features for sponsored search click prediction. In *AdKDD Workshop*, 2009.
34. R. Srikant, S. Basu, N. Wang, and D. Pregibon. User Browsing Models: Relevance versus Examination. In *KDD*, 2010.
35. B. Tan and F. Peng. Unsupervised query segmentation using generative language models and wikipedia. In *WWW*, pages 347–356, 2008.
36. D. Tang, A. Agarwal, D. O'Brien, and M. Meyer. Overlapping experiment infrastructure: more, better, faster experimentation. In *KDD*, pages 17–26, New York, NY, USA, 2010. ACM.
37. W. Xu, E. Manavoglu, and E. Cantú-Paz. Temporal click model for sponsored search. In *SIGIR*, 2010.
38. W. V. Zhang and R. Jones. Comparing click logs and editorial labels for training query rewriting. In E. Amitay, C. G. Murray, and J. Teevan, editors, *Query Log Analysis: Social And Technological Challenges. A workshop at the 16th International World Wide Web Conference (WWW 2007)*, May 2007.
39. Z. Zhang and O. Nasraoui. Mining search engine query logs for query recommendation. In *WWW*, 2006.
40. Z. Zheng, H. Zha, T. Zhang, O. Chapelle, K. Chen, and G. Sun. A general boosting method and its application to learning ranking functions for web search. In *NIPS*, pages 1697–1704, 2008.
41. D. Zhou, L. Bolelli, J. Li, C. L. Giles, and H. Zha. Learning user clicks in web search. In *IJCAI*, 2007.