Estimating Advertisability of Tail Queries for Sponsored Search

Sandeep Pandey Kunal Punera Marcus Fontoura Vanja Josifovski

Yahoo! Research
701 First Ave.
Sunnyvale, CA 94089
{spandey, kpunera, marcusf, vanjaj}@yahoo-inc.com

ABSTRACT

Sponsored search is one of the major sources of revenue for search engines on the World Wide Web. It has been observed that while showing ads for every query maximizes short-term revenue, irrelevant ads lead to poor user experience and less revenue in the long-term. Hence, it is in search engines' interest to place ads only for queries that are likely to attract ad-clicks. Many algorithms for estimating query advertisability exist in literature, but most of these methods have been proposed for and tested on the frequent or "head" queries. Since query frequencies on search engine are known to be distributed as a power-law, this leaves a huge fraction of the queries uncovered.

In this paper we focus on the more challenging problem of estimating query advertisability for infrequent or "tail" queries. These require fundamentally different methods than head queries: for e.g., tail queries are almost all unique and require the estimation method to be online and inexpensive. We show that previously proposed methods do not apply to tail queries, and when modified for our scenario they do not work well. Further, we give a simple, yet effective, approach, which estimates query advertisability using only the words present in the queries. We evaluate our approach on a real-world dataset consisting of search engine queries and user clicks. Our results show that our simple approach outperforms a more complex one based on regularized regression.

Categories and Subject Descriptors

H.3.m [**Information Storage and Retrieval**]: Miscellaneous

General Terms

Algorithms, Design, Experimentation

SIGIR'10, July 19–23, 2010, Geneva, Switzerland.
Copyright 2010 ACM 978-1-60558-896-4/10/07 ...$10.00.

Keywords

sponsored search, click estimation, tail queries

1. INTRODUCTION

Sponsored search is the dominant form of textual advertising on the Web in terms of revenue. It involves displaying advertisements (ads) alongside the results returned by search engines. Under the pay-per-click mechanism, search engines get paid every time a user clicks on a displayed ad. Clearly, sponsored search is useful for search engines since it is a source of revenue for them. Moreover, it is beneficial for users as well since it helps them in finding relevant products/services, especially for queries with commercial intent. It also aids advertisers in reaching the right set of users.

The success of sponsored search heavily relies on displaying relevant ads for appropriate queries. Previous studies [6], have shown that irrelevant or unwanted ads are useless to search engines since they do not attract clicks. They may even be harmful since they degrade the quality of search experience driving users away. Even users who continue using the search engine despite seeing irrelevant ads might get "trained" to ignore the sponsored sections of the search result page, impacting the long term revenue of the search engine. Hence, estimating the following two properties are of key importance for an advertising engine:

- **Query advertisability**: Certain queries are more suitable for advertising than others. For instance, queries such as "digital camera" and "car insurance" are more likely to be satisfied by sponsored search results than queries like "hotmail". Gauging the query advertisability correctly helps the advertising engine decide whether to show ads or not (or how many ads to display), and thus only showing ads to which the user will react. Moreover, this reduces the computation cost of ad selection for queries that should not display ads.
- **Ad relevance and clickability** : Once the advertising engine has determined that the query is advertisable, it attempts to retrieve the ads which are most likely to satisfy the user's information need. This typically involves ranking the ads in the system based on various factors like their syntactic match with the query, their estimated CTRs (click-through rates) from historical data, a user's past behavior etc. Several methods have been proposed for doing this ad selection and CTR estimation [4, 9, 14, 15, 18, 23, 24].

Estimating Advertisability for Tail Queries.

In this paper we focus on the first task above, that of estimating *Query Advertisability.* There is past work on identifying whether user queries have an underlying commercial intent, the intention to purchase a product or a service [2, 6, 10]. However, in addition to the underlying intent, advertisability should also capture other factors that influence the likelihood of engaging the user; the suitability of the current ad supply, the ability of the ad selection algorithms to select good ads, and even business rules. Therefore, we consider the ad clicks obtained in response to a query as a proxy for its advertisability. Specifically, in this paper we define advertisability of a query as the probability of seeing a click on any sponsored search ads displayed on the result page of the query.

Past work on modeling advertisability of queries have focused on using features derived from a plethora of information about them; the set of all retrieved ads in [6] and the set of all retrieved search results in [10]. Moreover, these approaches determine advertisability from an offline analysis of the historical click data. These methods have only been tested and shown to work well on frequently occurring queries. However, in this paper we focus on the more challenging problem of estimating advertisability of infrequent or *tail queries*, so called because they form the “heavy tail” of the power-law distribution of query frequencies on a search engine. The above mentioned approaches are not applicable to these tail queries as they are too rare to have significant historical data. Furthermore, tail queries are almost always unique, thus requiring an online estimation procedure (i.e., estimation is performed when users issue the queries). Since search users are very sensitive to any latency in the results presentation, under any reasonable system infrastructure the online procedure must be inexpensive and cannot employ complex query expansion methods. Hence we study the problem of estimating query advertisability using the query keywords only, similar to [2, 22].

Technical Challenges and Solutions.

Most of the queries in the datasets used in this study consist of 3-4 words and have occurred 1-2 times. This results in the two principal challenges of the problem: noisy ground truth due to the rarity of the queries, and sparseness of features due to the short query lengths.

The noise in the ground truth, i.e. the estimates of query advertisability, results from the low number of impressions for each query. This makes not only the learning difficult, but also affects the testing methodology. For instance, an oracle is also unlikely to match the advertisability estimates of individual queries derived using our data. One of our key insights is that though the advertisability of each individual query is noisy, when many queries are put together they provide a fairly stable advertisability estimate. Hence, given an estimation policy we evaluate it by looking at the aggregate advertisability of the top-ranked queries (instead of their individual advertisability).

In order to learn in the presence of noisy ground truth, we propose a word-based advertisability model that employs the above “grouping” insight (in Section 2). We estimate the parameters of this model via a maximum likelihood based method. As a competitive baseline we also present a regression based methodology that combines state-of-art elements from machine learning literature (in Section 3). Finally, orthogonal to these two methodologies we study an additional way of dealing with noise: learning from the head queries (which have reliable advertisability estimates) and then applying the model on the tail queries (see Section 2.2.1). From our experiments we found that this does not work well, since tail queries exhibit fairly different vocabulary and characteristics than the head queries.

We handle the sparsity of features in different ways for the two learning methodologies. For the maximum likelihood estimation method, we show that simplifying assumptions to remove interactions among the features result in improved accuracy. In the regression methodology, we handle sparsity using two methods. One way is to perform regression under a L^1-regularization (also known as Lasso). A second way is to make the features denser by clustering them [8, 22, 27]. In Section 3.3, we give an LDA-based method of clustering tail queries and a learning method that maps queries into latent clusters and learns a model using these cluster-based features.

Contributions.

We make the following contributions in the paper:

1) We investigate and formalize the problem of estimating advertisability of tail queries using its keywords only. The problem is challenging due to the inherent noise and sparsity present in the data.

2) We propose a simple, yet effective, word-based model to estimate query advertisability. Our estimation method is robust to noisy ground truth as well as sparse features.

3) We put together a competitive baseline regression-based approach that deals with sparsity by: (a) incorporating regularization in the model and (b) using LDA-derived latent topics.

4) We give an evaluation framework for the problem using a large scale dataset from a real-world search engine. Our extensive empirical results show that our word-based model outperforms the more expensive and complex regression-based approach.

2. WORD-BASED QUERY ADVERTISABILITY MODEL

Our goal in this work is to make predictions for Query Advertisability, which is the probability of the event when one or more ads displayed for a query are clicked. In this section, we give a model to accomplish this.

2.1 Model Formulation

In Section 1 we discussed the challenges of learning and evaluation in the presence of noisy ground truth and sparse features. Further, we used these to motivate our use of a word-based model. In this section, we start with a discussion of the additional properties that we want our word-based query advertisability model to have.

A basic desired property would determine the influence each word exerts on the advertisability of the query it is part of. Some words indicate that the user is looking for a certain product, e.g., “download”, “buy” and “compare”, while other words like “insurance”, “flight” and “hotel”, are associated with products/services that are known to be amenable to advertising. When we look at the queries that contain these words and their corresponding ad-clicks, we observe that these words have heavy positive influence on the query advertisability. Similar observations have been made in [2]. Hence, a useful basic property to have is, **P1**: a single "advertisable word" should be capable of ensuring high advertisability for a query containing it. On the flip side, consider some words from [2] that lead to low ad-clicks on queries, such as "weather", "free", "university" etc. From our data, we find that queries containing these words could potentially still be highly advertisable; for example, "weather in fiji", "free download", and "university admissions". Hence, a useful second property to have is, **P2**: while some words do not contribute to a query's advertisability, no one word's presence should reduce the advertisability. Finally, the effect of a word on a query advertisability might depend on other words present, like effect of "music" in the queries "music ringtones" and "music lyrics". However, because the word occurrences in tail queries are extremely sparse we do not incorporate such dependencies into the model.

Conforming to these desirable characteristics we give the following query advertisability model. In the model we say that each word in a query has a certain propensity of attracting a click on an ad, say $c(w)$. Let us denote the advertisability of query q by $c(q)$. Say, the query q consists of the words $w_1, w_2 \ldots w_n$. Thus, under the independence assumption, each word in the query independently attracts an ad-click for the query (with probability $c(w)$). Hence, the advertisability (i.e., the probability of the ads displayed for query q to be clicked) can be written as:

$$c(q) = 1 - \prod_{i=1}^{n} \big(1 - c(w_i)\big) \tag{1}$$

A key property of this formulation is that, all things being equal, it favors longer queries (e.g., if all $c(w_i)$'s were the same, $c(q)$ gets larger as n gets larger). While this makes sense in most cases, it can score longer queries containing words with mediocre click propensities higher than shorter ones with few good terms. To avoid this shortcoming, we introduce parameter k where k denotes the maximum number of words from the query that can take part in the clicking process. Under this constraint:

$$c(q) = \max_{\mathcal{S}} \left(1 - \prod_{w \in \mathcal{S}} (1 - c(w))\right) \tag{2}$$

where $\mathcal{S} \subseteq q$ and $|\mathcal{S}| \leq k$.

2.2 Parameter Estimation

In Equation 1 we presented the model to combine each query word's contribution to the query advertisability.[1] We can estimate the parameters of this model by computing the maximum likelihood estimate of the training data. The training data consists of queries and their associated click or impression events. Say, $s(q)$ denotes the number of instances when query q received an ad-click, while $n(q)$ denotes the number of instances when it did not. Given a dataset of queries $\mathcal{Q}$ and click events, its likelihood can be written as:

$$\begin{aligned}\mathcal{L}(s(q), n(q); c(w)) &= \prod_{q\in\mathcal{Q}} \Big(1 - \prod_{w\in q}(1 - c(w))\Big)^{s(q)} \\ &\times \Big(\prod_{w\in q}(1 - c(w))\Big)^{n(q)}\end{aligned}$$

On taking the logarithm of the both sides:

$$\begin{aligned}log\mathcal{L}(s(q), n(q); c(w)) &= \sum_{q\in\mathcal{Q}} s(q) \cdot log\big(1 - \prod_{w\in q}(1 - c(w))\big) \\ &+ \sum_{q\in\mathcal{Q}} n(q) \cdot log\big(\prod_{w\in q}(1 - c(w))\big)\end{aligned}$$

Taking derivatives with respect to $c(w)$ results in:

$$\frac{\sum_{q\ni w} n(q)}{(1 - c(w))} = \sum_{q\ni w} \Big(s(q) \cdot \frac{\prod_{\substack{w'\in q \\ w'!=w}} (1 - c(w'))}{1 - \prod_{w'\in q}(1 - c(w'))}\Big)$$

Here $q \ni w$ is the set of queries that contain the keyword w. Solving this complex set of equations is difficult, especially since the feature combinations used in queries are sparse and the ground truth is unreliable. Hence, we approximate this solution by assuming that each instance of a click or not click is a referendum on the advertisability of each keyword in the query independently. This has the same effect as replicating each query once for each term contained in it, with each such replication having just one of the keywords. Under this assumption we obtain:

$$c(w) = \frac{\sum_{q\ni w} s(q)}{\sum_{q\ni w}(s(q) + n(q))} \tag{3}$$

In other words, the contribution of a word to advertisability is the fraction of times it is present in a query instance which attracted an ad-click.

2.2.1 *Choice of Training Set*

As mentioned earlier, tail queries have very few impressions, the words combinations are extremely sparse, and thus their advertisability estimates tend to be very noisy. Clearly, learning from such a dataset is difficult and may lead to a poor model estimation.

An alternative approach is to learn the model from the head queries, which tend to have significant number of occurrences in the historical data. For these queries we can compute advertisability estimate with confidence. The disadvantage of learning from such a dataset is that it is very different from the test set of tail queries in terms of vocabulary, word combinations, and maybe even word advertisability scores. This difference could counteract the advantages gained from reliable ground truth when learning on the head queries.

Each training set has its own advantages and disadvantages. While the training set of tail queries is noisy, the training set of head queries may exhibit different behavior and not generalize well on the test set. In Section 4.5 we evaluate our model while training on both the head and tail set of queries.

3. REGRESSION-BASED QUERY ADVERTISABILITY MODEL

[1] In Equation 2 we use the k "best" terms to compute query advertisability, but we will use Equation 1 to estimate the model parameters.

In Section 2, we proposed a simple word-based model for predicting query advertisability. In this section, we present alternative approach that combines state of the art elements from machine learning literature. First, we will formulate the task of predicting query advertisability as a regression problem. Then we will present a few ideas to help the regression model handle sparsity: regularization and clustering.

3.1 Linear Regression Model

Just as in the word-based model in Section 2, we say that each word in a query has a certain propensity of attracting a click on an ad; we denote this with $c(w)$. The advertisability of query q, denoted by $c(q)$, can then be computed quite naturally by summing of the individual word advertisability values [19, 20]. Under this model we consider words in queries as binary features and the weight of each word is given by its advertisability. Say, the query q consists of the words $w_1, w_2 \ldots w_n$, then $c(q) \approx \sum_{w \in q} c(w)$.

Hence, we can write the following set of linear equations:

$$\forall q, \sum_{w \in q} c(w) = c(q) + \epsilon_q$$

where ϵ_q's are the error terms.

Under the squared loss function this problem can be formulated as:

$$\arg\min_{c(w)} \sum_{q \in \mathcal{Q}} \Big(c(q) - \sum_{w \in q} c_w \Big)^2$$

This set of linear equations can be solved using existing methods. However, we face another challenge here which is that due to the sparsity in word occurrences. A tail query consists of 3-4 words on average, which is a very little amount of text. Moreover, tail queries significantly differ from each other, thus resulting in a large vocabulary of words. For instance, in our experiments we noted that the number of unique words is more than half of the number of unique queries. In other words, if we represent a query in this feature space, it will consist of a couple of non-zero entries for the words present in the query, while the rest of the thousands of dimensions will be all zero. This is likely to make this set of equations under-determined. Next we consider a couple of ways to handle this issue of sparsity.

3.2 Regularized Regression

When the number of parameters is large, the estimates of linear regression exhibit high variance which is undesirable. One way of controlling this by incorporating regularization while training. Under L1 regularization (also known as Lasso [12, 26]) this can be written as:

$$\arg\min_{c(w)} \sum_{q \in \mathcal{Q}} \Big(c(q) - \sum_{w \in q} c_w \Big)^2$$

subject to:

$$\sum_{w} c(w) \leq t$$

where λ is a given constant. This is also written as:

$$\arg\min_{c(w)} \sum_{q \in \mathcal{Q}} \Big(c(q) - \sum_{w \in q} c_w \Big)^2 + \lambda \sum_{w} c(w)$$

where λ is the shrinkage parameter. When λ is close to 0, this behaves like regular linear regression, while as λ goes to infinity it forces many $c(w)$'s to be zero. Hence, this performs feature selection for us, though unlike traditional feature selection methods this is not limited to completely picking or dropping a feature.

3.3 Inferring Topics from Queries

An alternative method of dealing with sparsity is by mapping the sparse high-dimensional feature space to a dense low-dimensional space. Principal component analysis is often used in doing so while maximizing the variance of the data captured in the low-dimensional space [17]. Latent semantic analysis is also used for dimension reduction [11, 16]. It transforms the sparse word-document matrix to a more dense topic-document matrix, where each topic is a latent concept that is derived using the co-occurrence information.

In this paper, we use Latent Dirichlet Allocation to obtain topics from queries. This is a generative model for the documents where the topic distribution is assumed to have a Dirichlet prior [5]. We describe it in more details next.

3.3.1 Latent Dirichlet Allocation

In this model a document is assumed to be generated from a mixture of topics, where each topic has its own word distribution. In particular, we can write the probability of the i^{th} word in the document as (given in [13]):

$$P(w_i) = \sum_{j=1}^{T} P(w_i | z_i = j) P(z_i = j)$$

where T is the number of topics. Variable z_i denotes the latent topic from which word w_i is generated. It is equal to topic j with prior probability $P(z_i = j)$, in which case the word has $P(w_i | z_i = j)$ of being generated given the word distribution of the j^{th} topic.

Let T denote the number of topics and D denote the number of documents. In the LDA model $P(w|z)$ is modeled using a set of T multinomial distributions ϕ over the vocabulary W (one multinomial ϕ per topic). The ϕ distribution of a topic gives the distribution of words under the topic. $P(z)$ is modeled using a set of D multinomial distributions, denoted by θ^d, over T topics (one multinomial θ per document). The multinomial distribution θ^d gives the mixture of T topics present in the document d, i.e., $\theta_j^{(d)} = P(z = j)$. Both θ and ϕ distributions have Dirichlet priors [13]. In brief, the model can be written as:

$$\begin{array}{lcl} w_i | z_i, \phi^{(z_i)} & \sim & \text{Discrete}(\phi^{(z_i)}) \\ \phi & \sim & \text{Dirichlet}(\beta) \\ z_i | \theta^{(d_i)} & \sim & \text{Discrete}(\theta^{(d_i)}) \\ \theta & \sim & \text{Dirichlet}(\alpha) \end{array}$$

where α and β are the hyperparameters.

3.3.2 Deriving LDA Topic-based Features

We use LDA to find a low dimensional representation of a query. In particular, we run LDA over a training set of queries for T number of topics. After the end of run, we have a topic distribution of each query q, $P(z|q)$. Each topic makes a latent concept and the topic distribution of query $P(z|q)$ gives a representation of the query in this low-dimensional concept space. For each query, these posterior topic-membership values can be used in two ways: they can be added to the query words as additional binary features, or used as a sole representation of a query. In this paper we experiment with both methods. We can use these query representations in our model from Section 3.1. In particular, we find the advertisability of each topic using regularized regression and use them to compute the advertisability of a query.

The advantage of this method is that it is able to relate query keywords with each other in an intelligent manner. For example, we found that in our experiments a topic consisted of words like "ipod", "iphone", "samsung" etc. So, using just the co-occurrence information of data, LDA was able to connect these words together which are clearly very related. This helps significantly since new queries that fall into this cluster will be able to use the advertisability information estimated for all queries that fall into the cluster, hopefully leading to robust results. In Section 4.3 we empirically evaluate this approach.

4. EXPERIMENTS

In this section we evaluate the word-based model we proposed for query advertisability with the state-of-art regression based methodology.

4.1 Empirical Setup

Dataset.

We train and evaluate our approach on a real-life search engine data. We collected a sample of queries issued to a major search engine over a period of 7 days. The dataset consists of more than 5 million query impressions. We also recorded the ad clicks for these queries during this period. Of these queries we put those queries into the *tail set* which have less than 2 impressions per day on average. The goal of this study is to predict the advertisability on these tail queries. The aforementioned tail set consists of more than 2 million unique queries. For our experiments we placed half of the queries in the training set and the rest in the test set.

The average query length is 3.3 which shows the amount of sparsity in the data. The average number of impressions per query is 1.55. As mentioned earlier, given such a few impressions for a query it is difficult to estimate its advertisability with any certainty. This presents challenges while learning as well as evaluation. In view of this, we propose our evaluation metric next.

Evaluation Metric.

A conventional way of evaluating our advertisability estimation methods would be to take the L^1-error or L^2-error of the predicted ($\hat{c}(q)$) and "true" advertisability of queries ($c(q)$). However, since our focus is on tail queries which have very few impressions (as shown above), it is not possible to estimate their true advertisability. In other words, given the noise in "true" advertisability estimated from our data an oracle is also unlikely to match them.

To deal with this problem, we opt to evaluate the prediction of a method with respect to a group of queries instead of the individual queries. In particular, given method π we rank the queries in order of their decreasing predicted advertisability values $\hat{c}(q)$'s. Starting from the top, let $S(r,\pi)$) and $F(r,\pi)$ be the cumulative total of number of ad-clicks and the total number of all impressions till rank r. Due to aggregation over a group of queries, $S(r,\pi)$ and $F(r,\pi)$ are fairly stable for large values of r and amenable for conducting analysis. Clearly, the best method is the one which maximizes $S(r,\pi)$ for all values of $F(r,\pi)$. In case there is no clear best method, we can plot $S(r,\pi)$ on the y-axis and $F(r,\pi)$ on the x-axis and compute area under the curve (AUC) to succinctly summarize the performance of a method. The best policy for a given impression threshold, say τ, is the one which maximizes $S(r,\pi)$ for $F(r,\pi)=\tau$.

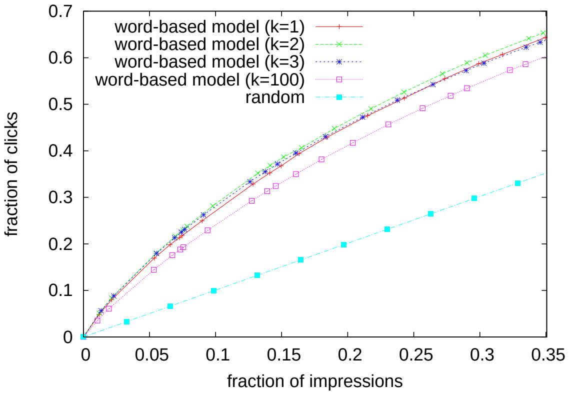

Figure 1: Performance of the word-based advertisability model for different values of k.

Another reason this evaluation method is suitable is that launch criteria in real-world systems are likely to be framed w.r.t. to the relationship between $S(r,\pi)$ and $F(r,\pi)$. In order to preserve the user's search experience, an advertising system controller will likely limit the number of instances of queries in which ads are shown. The controller would then be interested in determining the algorithm that yields the most clicks in the fixed number of impressions. On the flip side, the number of clicks might be fixed in order to make revenue numbers.

4.2 Evaluation of Word-based Advertisability Model

In this experiment we evaluate the performance of our word-based advertisability model presented in Section 2. We tokenize each query into words by treating whitespace for word boundaries. Since many of our queries are URLs, we tokenize these queries at punctuation characters. We remove the stop-words and stem the remaining words.

For this experiment we learn word advertisability scores ($c(w)$'s) from the training set using the approximate method (Equation 3) of Section 2.2. Then we evaluate the method on the test set. Figure 1 shows the performance of our method for different values of k. Recall that k is a parameter in our method which limits the number of words from a query that can contribute towards its advertisability. The x-axis in the figure is the cumulative fraction number of impressions till a given rank ($F(r,\pi)$), while the y-axis is the cumulative fraction of clicks ($S(r,\pi)$).

Note that all of our model variants are significantly above the straight line which denotes the random method (i.e., predict the advertisability at random). This is encouraging since it shows that the advertisability can be estimated quite well using the query keywords only. As expected the curves in the figure are convex. This happens because when the number of impressions is small (x-axis), the clicked impressions (y-axis) are aggregated over queries present on the top of the ranked list. These queries are predicted to have high advertisability, thus resulting in a high average click per impression (i.e., the slope of the curve). However, as the number of impressions increases, queries with low advertisability start getting accounted for and hence, they bring down the average click per impression.

From the figure it is clear that $k = 2$ performs the best. Intuitively, this makes sense because when k is too small, the method does not give due credit to queries with more advertisable words. On the other hand, when k is too large, long queries get an unfair advantage. Still, the model looks fairly stable when k is in a reasonable range (i.e., 1 to 3).

4.3 Evaluation of Regression-based Approach

In Section 3 we gave a regression-based approach to predict query advertisability. Due to sparsity the simple linear regression is unlikely to work, hence we experiment with the L^1-regularized regression. We perform L^1-regularization using the SMIDAS software [25] where SMIDAS stands for “Stochastic Mirror Descent made Sparse”.

Furthermore, as discussed in Section 3.3.2, we use LDA to derive dense topic-based features. In particular, we represent each query as a mixture of latent topics derived using LDA and the query words. We then learn the model over this hybrid low-dimensional query representations using the SMIDAS software as above. In Table 2 we show some topics and the top words in them as derived using LDA. Clearly, the topics are fairly coherent and meaningful. We experimented with constructing a 100 and a 1000 topics; the results were very similar and we present the ones with 100 topics. We also experimented with using just the low-dimensional representations of a query in the regression formulation, but the results were much worse; we do not present those results here.

In Figure 2 we plot the performance of the regression-based approach with and without the topic-based features. As we can see adding the topic-based features has so significant effect on the accuracy of the prediction task. This is in contrast to the results obtained in [22], where keyword cluster based features were shown to have significant impact on accuracy. We posit that this is because our focus is on tail queries; while the topics constructed by LDA are themselves meaningful, the uniqueness of tail queries means that inferring the topics for each tail query is extremely difficult. Hence, the knowledge contained in sparse word-based features subsumes the contribution of dense topic-based features. We leave further investigation of this phenomena for future work.

4.4 Comparison of Word-based and Regression-based Models

Above we evaluated the two different methods for estimating query advertisability. In Table 1 we show the popular words with top advertisability under the two methods. Note that both methods are doing a reasonably good job of finding highly advertisable words such as rental, vacation, travel etc. Even though the top features from the two estimation methods look fairly similar, the weights of many other features show differences.

In particular, we plot the performance of the two approaches, word-based approach and L^1-regularized regression, in Figure 3. It is clear that the word-based method performs better in comparison to its state-of-art counterpart. This shows that careful modeling of the properties of the problem can result in a simple, yet effective, approach that can outperform a method based on much more complex machine learning primitives.

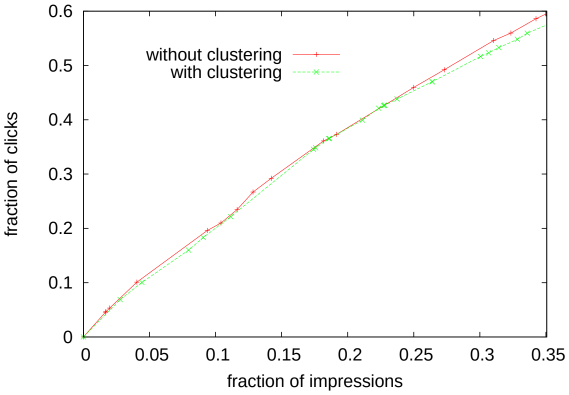

Figure 2: Performance of the regression-based approach with and without LDA-derived topic features.

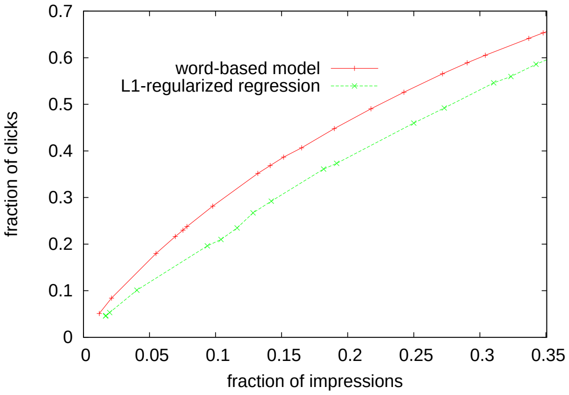

Figure 3: Performance comparison of the word-based model and regression-based approach.

4.5 Choice of Training Set

So far we have been learning the model from the training set of tail queries. Another data set that can be employed for training is the head set which consists of all the head queries. The advantage of this set is that it consists of frequent queries and has relatively stable query advertisability values ($c(q)$). The disadvantage is that it is not similar to the test set which consists of tail queries only.

We perform learning from these training sets using model-based method for $k = 2$. Figure 4 shows the performance for different training sets. Note that the training set of head queries performs worse. This shows that the naive approach of learning from head queries and applying the learned model on tail queries does not work well since they exhibit different characteristics than the head queries. This does not imply that head queries are useless for our task; instead, it means that head queries are not sufficient by themselves and must be used in conjunction with tail queries to aid the learning.

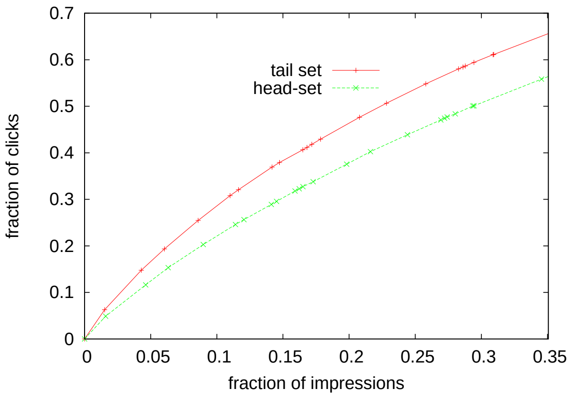

Figure 4: Performance of word-based model under different training sets.

Method	Top Words
Word-based Model	cheap, boot, ticket, discount, bag, bed, laptop, wholesale, chevrolet, airline, rent, flight, coupon, shower, rental, nike, vacation, loan, hotel, furniture, diet, cruise, outlet, dress, job, phone, printer, hp, truck, car, price, dvd
L^1-regularized Regression	chair, costume, store, hotel, wholesale, diet, dress, rental, boot, camera, part, sale, cheap, price, inn, bed, ticket, travel, rent, shower, cruise, bathroom, batteries, ring, vegas, furniture, car, free, shoes, curtain, loan, discount

Table 1: Words with high advertisability estimates under the two prediction methods.

Topic	Words
1	free, game, online, download, video
2	honda, toyota, bmw, yamaha, tire
3	ipod, iphone, samsung, sim, touch
4	truck, chevy, chevrolet, engine, parts
5	diet, product, pill, lose, fat, weight
6	coffee, table, top, bed, chair, antique
7	lyrics, love, song, victoria, secret
8	coupon, pizza, code, family, restaurant
9	county, court, sheriff, public, office, clerk
10	job, office, apply, description, salary
11	phone, number, service, verizon, sprint
12	loan, money, student, mortgage, finance, grant, aid
13	california, sacramento, pittsburgh, riverside, buffalo
14	army, base, navy, military, force, nation
15	digital, camera, review, batteries, samsung
16	lose, weight, blood, pressure, sugar, diabetes, skin, pill
17	sony, camera, digital, photo, memory, screen, film, batteries
18	hawaii, beach, resort, ski, spa, car, rental, hotel, vacation
19	software, printer, dell, monitor, screen, driver, laptop, computer
20	phone, service, motorola, verizon, sprint, cell, mobile, wireless, call

Table 2: Some LDA-derived topics and their top words.

5. RELATED WORK

Sponsored search is an active area of research. Several studies have been published recently that focus on the sponsored search advertising [1, 6, 7, 21, 22, 24]. We classify the related work along the following aspects and distinguish our work from them.

Modeling Ad-specific CTR.

Most prior art [7, 21, 22, 24] in sponsored search deals with estimating the CTR of a given query-ad pair; this estimate is often used to display the ads with the highest predicted CTR for a given query. In this work, we are interested in predicting the advertisability of queries, which we define as the probability of seeing an event in which one of the ads displayed for the query gets clicked. Gauging the query advertisability correctly helps the advertising engine decide whether to show ads or not (or how many ads to display), and thus only showing ads to which the user will react. Moreover, this reduces the computation cost of ad selection for queries that should not display ads.

Modeling Commercial Intent of User Queries.

Another line of related work has focused on identifying queries with commercial intent. An approach for detecting the commercial intent is proposed in [10]. They define the term OCI (Online Commercial Intention) and present a framework of building machine learning models to learn OCI based on the Web page content. They use that framework to detect the commercial intent of queries, which is related to the problem we solve in this paper. In [3] the authors analyze the click-through behavior of ads to characterize and predict query intent. An analysis of the contributions of the different query terms and their corresponding click rates on commercial intent queries is presented in [2]. In a follow-up work [1] the authors examine detecting commercial intent by building a classifier based on editorial judgments of the commercial intent of the queries. They show that those queries that are characterized as commercial have higher CTR than the others.

Our work differs form this set of works in a few ways. First, we focus on modeling query advertisability, which in addition to the underlying intent, also captures all the factors that influence the likelihood of engaging the user; the suitability of the current ad supply, the ability of the ad selection algorithms to select good ads, and even business rules. Because of this we consider the ad clicks obtained in response to a query as a proxy for its advertisability. Hence, unlike past work that has relied on using human judgments for learning, in our approach we use the click data directly, without using a human understandable definition of the property of interest. Thus we identify directly the queries for which the users would be inclined to click on ads. Last, we focus specifically on the more challenging problem of modeling for tail queries. These queries, due to their rarity and the sparseness of their term combinations present very different problems than considered in past work. In this paper we give some solutions to these problems.

Modeling using Rich Query Features.

In most past work, modeling of commercial intent behind queries has focused on using features derived from a plethora of information about them; the set of all retrieved ads in [6] and the set of all retrieved search results in [10]. Moreover, these approaches determine advertisability from an offline analysis of the historical click data. These methods have only been tested and shown to work well on frequently occurring queries. However, in this paper we focus on the more challenging problem of estimating advertisability of tail queries. The above mentioned approaches are not applicable to these tail queries since they are too rare to have significant historical data. Furthermore, tail queries are almost always unique, thus requiring an online estimation procedure (i.e., perform the estimation when users issue the queries). Since search users are very sensitive to any latency in the results presentation, under a reasonable system infrastructure the online procedure must be inexpensive and cannot employ complex query expansion methods. Hence we study the problem of estimating query advertisability using the query keywords only.

Regression with Cluster-based Features.

Our work is most close to the work in [22], where the authors propose to the clustering of the bid phrases of ads in estimating CTRs. Both top-down and bottom up hierarchical clustering are applied. The CTR of a bid phrase is then calculated as a linear combination of the predicted CTR and the CTR of its cluster. The results show that the smoothing helps the estimates for rare bid phrases and it slightly decreases the precision for common bid phrases. In this paper we update this approach by performing a more sophisticated topic modeling using LDA, and feeding the results of it into a state of the art learner based on regularized regression. We show that our our simple, yet effective, word-based model outperforms this regression/clustering based approach via empirical results in Section 4.

6. SUMMARY

In this paper we focused on the problem of estimating the advertisability of tail queries. Furthermore, due to some exogenous practical constraints, we performed the estimation using the query keywords only. We discussed how noisy ground truth and sparsity in the data make this problem difficult and techniques from past work do not apply well to our scenario. We showed how to deal with problems associated with tail queries by proposing a words-based query advertisability model. We gave a maximum likelihood method of learning the model. We also gave a competitive baseline methodology based on a regression formulation of the problem that used state-of-art machine learning approaches to deal with sparsity of data: (a) incorporating L^1-regularization in the model training and (b) finding latent topics using LDA. We conducted extensive experiments on real data to evaluate our model. Our results are encouraging and show that the advertisability of queries can be estimated pretty accurately using their keywords only. We also compared different model estimation methods and study the effect of regularization, clustering, and training set selection.

7. REFERENCES

[1] A. Ashkan and C. Clarke. Characterizing commercial intent. In *CIKM*, 2009.

[2] A. Ashkan and C. Clarke. Term-based commercial intent analysis. In *SIGIR*, 2009.

[3] A. Ashkan, C. Clarke, E. Agichtein, and Q. Guo. Characterizing query intent from sponsored search clickthrough data. In *SIGIR Workshop*, 2008.

[4] A. Ashkan, C. Clarke, E. Agichtein, and Q. Guo. Estimating ad clickthrough rate through query intent analysis. *Web Intelligence and Intelligent Agent Technology, IEEE/WIC/ACM International Conference on Web Intelligence*, 1:222–229, 2009.

[5] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[6] A. Broder, M. Ciaramita, M. Fontoura, E. Gabrilovich, V. Josifovski, D. Metzler, V. Murdock, and V. Plachouras. To swing or not to swing: Learning when (not) to advertise. In *CIKM*, 2008.

[7] 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*, 2009.

[8] J. Carrasco, D. Fain, K. Lang, and L. Zhukov. Clustering of bipartite advertiser-keyword graph. In *ICDM*, 2003.

[9] D. Chakrabarti, D. Agarwal, and V. Josifovski. Contextual advertising by combining relevance with click feedback. In *WWW*, 2008.

[10] H. Dai, L. Zhao, Z. Nie, J.-R. Wen, L. Wang, and Y. Li. Detecting online commercial intention (OCI). In *WWW*, 2006.

[11] S. Deerwester. Improving information retrieval with latent semantic indexing. In *Proceedings of the 51st Annual Meeting of the American Society for Information Science*, pages 36–40, 1988.

[12] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32:407–499, 1996.

[13] T. Griffiths and M. Steyvers. Finding scientific topics. *PNAS*, 101, 2004.

[14] W. Guo and G. Li. Predicting click rates by consistent bipartite spectral graph model. In *Proceedings of the 5th International Conference on Advanced Data Mining and Applications*, 2009.

[15] M. Gupta. Predicting click through rate for job listings. In *WWW*, 2009.

[16] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR*, 1999.

[17] I. Jolliffe. *Principal Component Analysis.* Springer-Verlag New York Inc, 1986.

[18] A. Lacerda, M. Cristo, M. Goncalves, W. Fan, N. Ziviani, and B. Ribeiro-Neto. Learning to advertise. In *SIGIR*, 2006.

[19] W. Mendenhall and T. Sincich. *A Second Course in Statistics: Regression Analysis.* Pearson Education, 2003.

[20] D. Montgomery, E. Peck, and G. Vining. *Introduction to Linear Regression Analysis.* New York: Wiley, 2001.

[21] 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'08*, 2008.

[22] M. Regelson and D. Fain. Predicting click-through rate using keyword clusters. In *Proceedings of the Second Workshop on Sponsored Search Auctions*, 2006.

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

[24] M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: Estimating the click-through rate for new ads. In *WWW*, 2007.

[25] S. Shalev-Shwartz and A. Tewari. Stochastic methods for l1 regularized loss minimization. In *ICML*, 2009.

[26] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1996.

[27] J. Yi and F. Maghoul. Query clustering using click-through graph. In *WWW*, 2009.