Ad Impression Forecasting for Sponsored Search

Abhirup Nath†, Shibnath Mukherjee‡, Prateek Jain†, Navin Goyal†, Srivatsan Laxman†
t-abhin, shibnatm, prajain, navingo, slaxman}@microsoft.com
†Microsoft Research, Bangalore, India
‡Microsoft adCenter, Bangalore, India

ABSTRACT

A typical problem for a search engine (hosting *sponsored* search service) is to provide the advertisers with a forecast of the number of impressions his/her ad is likely to obtain for a given bid. Accurate forecasts have high business value, since they enable advertisers to select bids that lead to better returns on their investment. They also play an important role in services such as automatic campaign optimization. Despite its importance the problem has remained relatively unexplored in literature. Existing methods typically overfit to the training data, leading to inconsistent performance. Furthermore, some of the existing methods cannot provide predictions for new ads, i.e., for ads that are not present in the logs. In this paper, we develop a generative model based approach that addresses these drawbacks. We design a Bayes net to capture inter-dependencies between the query traffic features and the competitors in an auction. Furthermore, we account for variability in the volume of query traffic by using a dynamic linear model. Finally, we implement our approach on a production grade MapReduce framework and conduct extensive large scale experiments on substantial volumes of sponsored search data from Bing. Our experimental results demonstrate significant advantages over existing methods as measured using several accuracy/error criteria, improved ability to provide estimates for new ads and more consistent performance with smaller variance in accuracies. Our method can also be adapted to several other related forecasting problems such as predicting average position of ads or the number of clicks under budget constraints.

General Terms

Algorithms, Economics, Forecasting

Keywords

Sponsored search, Auctions, Bayes net, Dynamic Linear Model

1. INTRODUCTION

Sponsored search has become an important channel of efficient online advertisement and is a multi-billion dollar industry today. It provides value to advertisers and users by providing targeted advertising, and is the major source of revenue for search engines.

WWW 2013, May 13–17, 2013, Rio de Janeiro, Brazil.
ACM 978-1-4503-2035-1/13/05.

Figure 1: An overview of the sponsored search process: When a user searches for a string say "Hotels", the search engine sends the extracted keyword (from the query string) for an auction between advertisers, who are willing to pay certain amount of money for a click by the user.

Each time a user issues a search query on a web search engine such as Google or Bing, an auction is invoked among advertisers who bid for the search query phrase and the winning ads appear alongside the corresponding 'organic' search results. See Figure 1 for an overview of the sponsored search process.

The auctions are conducted amongst advertisers where winners are determined according to their scores given by: $\text{Score}(L) = \text{bid}(L) \times \text{pclick}(L, Q)$, where $\text{Score}(L)$ is the score of ad L for the auction of query Q, $\text{bid}(L)$ is the bid of L, and $\text{pclick}(L, Q)$ is the predicted Click Through Rate (CTR), i.e., the probability that ad L is clicked when shown to the user in response to query Q. Typically, CTR (or $\text{pclick}(L, Q)$) is estimated by using a learning algorithm such as the one proposed in [5]; these algorithms continuously adjust their estimates of pclick according to the observed click patterns.

Sponsored search is a dynamic process where the set of advertisers and ads change continuously, and the query traffic demonstrate substantial temporal, seasonal and geographic variations (e.g. holiday season, events in news etc. can lead to big fluctuations in traffic volumes of specific queries). This makes it hard for advertisers to set bid values to achieve their objectives (e.g., advertisers may want to maximize the number of impressions/clicks for a given budget). Many advertisers, especially small-scale ones, do not set bids effectively and often end up with left-over budgets. A tool that forecasts the number of impressions, average position and clicks for an ad, given a bid value would therefore be of very useful to advertisers. An estimate of the number of impressions, together with the pclick

estimate using methods like [5] would give us an estimate for the number of clicks for the ad. Thus, one of the main challenges that needs be addressed is accurate estimation for the number of impressions of an ad given its bid value. This is the problem we focus on in this paper.

The ad impressions forecasting problem is challenging for the following reasons: (1) Query traffic can be highly variable due to multiple reasons as previously discussed. (2) The set of advertisers and ads changes with time. Advertisers can change their bids; budget pauses and budgets running out are another source of uncertainty. (3) Pclick of an ad varies from auction to auction because of changing query features (e.g. time and location of the query), and also because the pclick estimation algorithm itself can introduce large variations.

Search engines generally have tools that replay past auctions with changing bids and show the result to the advertisers as a proxy for future estimates. This type of past replay makes an inherent simplifying assumption that the auctions in the future are exact replicas of the auctions in the logs. Because of the aforementioned challenges, these assumptions deviate widely in practice leading to errors and inconsistency due to heavy overfitting. There are also a few recent works addressing similar problems, e.g. [1, 3, 8]; we discuss these in some detail in Section 3. These works focus mostly on modeling how advertisers bid in auctions, and do not explicitly model the query traffic. As we show in Section 5 query traffic modeling is a critical component in impression prediction and ignoring it can lead to poor and inconsistent forecast.

In this work, we address the aforementioned challenges by taking a learning-centric view of the problem: our method models auctions by a generative model to avoid overfitting. Specifically, we use a Bayes net to model the query traffic component as well as the minimum score required to win the auction. Since most of the auction features are categorical or can be efficiently discretized, Bayes net can be easily trained and sampled to generate artificial auctions. Furthermore, we parallelize Bayes net training using MapReduce; this parallelization is critical for deployment in real-life large scale systems.

For prediction, we generate artificial auctions using our Bayes net. We then assess win or loss for the ad in each auction to determine the total number of impressions.A crucial issue here is the estimation of the number of samples to be generated from Bayes net. We use a first order dynamic linear model trained on past keyword traffic trends to this end.

We conduct experiments on a substantial portion of traffic from Bing [1] and evaluate performance of our method against two well-known existing methods according to several criteria. Our empirical results demonstrate significantly improved and consistent forecasts on multiple criteria compared to the existing methods. Specifically, our method achieves up to 20% more accuracy than the existing methods and can predict for up to 33% more ads than one of the baseline methods. Furthermore, our method is highly scalable and can be deployed in a real-life system. Our offline training phase takes about four to five hours using standard production architecture, while prediction for advertisers can be provided in real time (online) using pre-computed impression values.

2. PRELIMINARIES AND NOTATION

In this section, we formally introduce the setting of sponsored search auctions.

Following standard terminology, we refer to an ad in sponsored search setting as a listing and denote it by L. Traffic features of a user query are denoted by Q; this includes features such as location, category, and time of the query. $\text{bid}(L)$ denotes the bid value of listing L and $\text{pclick}(L, Q)$ denotes the estimated probability of click on listing L when a user makes query with features Q.

In sponsored search, the ad serving engine conducts an auction for every user query. For each such query phrase, advertisers compete with pre-set bids and the score for each ad L is given by:

$$\text{Score}(L, Q) = \text{bid}(L) \times \text{pclick}(L, Q). \quad (1)$$

$\text{pclick}(L, Q)$ is estimated using a learning algorithm that uses several features from the query Q such as the traffic features mentioned above and also features from the listing L and the associated advertiser; see [5] for more details.

After computing the score of each listing for a given query, the scores are sorted and at most k listings with highest scores are selected, where k is a parameter set by the search engine and can vary across auctions. Out of these k, only those with scores greater than a reserve score (again a parameter set by the search engine) are selected. There may be additional criteria to further prune this list of ads. If an advertiser's ad is clicked by the user in auction Q, then the advertiser makes a payment to the search engine. This payment is calculated by the GSP method [4]:

$$\text{Payment}(L_i, Q) = \frac{\text{Score}(L_{i+1}, Q)}{\text{pclick}(L_i, Q)},$$

where L_{i+1} is the listing after L_i in the sorted scores list.

3. RELATED WORK

Ads impression forecasting in sponsored search has recently become an important tool, hosted by search engines. The goal is to help advertisers bid appropriately to achieve their return over investment (ROI). Although, the tool is heavily used in practice, there has been little research addressing this problem.

Notable exceptions are the works by Athey and Nekipelov [1] and Pin and Key [8]. Both of these papers assume a probabilistic model on the given ads' competitors' score. They learn parameters of such model using training data and generate and simulate auctions to determine number of impressions. In particular, Pin and Key [8] assume that the "normalized scores" ($ns(L', Q) = \text{bid}(L') \times$ $\text{pclick}(L', Q)/\text{pclick}(L, Q)$) of all the competitor ads L' are sampled i.i.d. from a fixed distribution (where L is the ad for which we are doing the prediction). Now to predict the slot in which L will appear in an auction, we just need to check for how many ads L', we have $\text{bid}(L) < \text{bid}(L') \times \text{pclick}(L', Q)/\text{pclick}(L, Q)$ (which is same as $\text{bid}(L) \times \text{pclick}(L, Q) < \text{bid}(L')\text{pclick}(L', Q)$, i.e., $\text{Score}(L) < \text{Score}(L')$). For a given bid $\text{bid}(L)$, one can then derive the expected number of impressions and clicks. Note, however, that there are issues with the assumptions in this model: For instance, the assumptions that the scores have the same distribution across auctions do not hold in practice. Normalized scores have $\text{pclick}(L, Q)$ in the denominator, and $\text{pclick}(L, Q)$ tends to be highly variable, thus affecting all the normalized scores and introducing correlations.

The Athey and Key [1] model is similar but more involved and detailed and also computationally more demanding. [1] show that the distributional assumptions actually lead to simplification in the set of equilibria (compared to the results in [4, 9]) and can even lead to unique equilibrium under certain conditions. [1, 8] can also infer advertiser's value per click assuming that they bid optimally. In accuracy, [1] does just slightly better.

Duong and Lahaie [3] use discrete choice analysis, a technique developed in econometrics, for the problem of inferring advertiser's

[1]Exact percentages are not provided for confidentiality reasons.

value per click. Using the estimated values they can predict how many clicks and impressions an ad will receive in the near future. They, however, require that the ad in question be present in the training week. Their experimental results show accuracy somewhat comparable to [8] in predicting the number of clicks.

Recently, similar problems have also been addressed in the other online advertising paradigms, namely, contextual and display advertising. For contextual ads, Wang et al. [10] propose a method for impression forecasting based on replay of past auctions with the given advertisement. When adapted to sponsored search domain, their approach reduces to *Training Week Replay* (TWR); we empirically evaluate our method against TWR (see Section 5). Also note that, the efficient search strategies proposed by [10] do not apply to our problem as the number of "published" pages are significantly larger for sponsored search than for contextual ads domain.

For display ads, Cui et al. [2] propose a method for impression forecasting based on Gradient Boosted Decision Trees. However, the display ads and sponsored search settings are significantly different: In the display ads setting there are a small number of possible "targettings" for which bid landscape needs to be learned, hence there is enough data to learn a regression model for each parameter setting. In contrast, in sponsored search, the number of possible "parameters" or "targettings" is extremely large (due to large number of query strings and also large number of contexts that give rise to the query strings), hence a reliable regression model cannot be learned for each parameter.

Further pointers to the literature can be found in the above cited papers. In this paper, we will show that our result compare favorably with those of [8]. This will also mean that our results compare favorably with those of [1, 3].

4. METHODOLOGY

In this section, we first describe the problem of ad impression forecasting and then provide our proposed method for this problem.

Ad impression forecasting problem: Given an advertiser A and its advertisement (or listing) L, and a bid value bid(L), the goal of ad impression forecasting is to predict the number of impressions L is likely to obtain in a fixed amount of time in future. For simplicity of exposition and for several practical reasons, we assume that the prediction is made for next 1 week [2]. The training data available for our forecasting problem are auction logs that contain information about all auctions from the recent past, such as, query traffic features, scores of the winners etc.

For the above mentioned problem, our method learns a generative models for auctions, in which the given listing L participates. For modeling, we view an auction (in which the given listing L participates) as a tuple of: 1) features of the query for which the auction is held, 2) competitors' scores, 3) score for the given listing L. Below, we further explain the above mentioned aspects of an auction that we model:

- **Query**: Search engines typically extract several features from a user query. For example, features can include the location where query originated from, time of the query, category of the query, keywords in the query etc. These query traffic features form an important component of auctions because both competitors' score as well as listing L's score depend heavily on these features. For example, suppose listing L got several clicks from users in New York. Hence, if the query originates from New York then the pclick of L would tend to be high, leading to high score for L in those auctions.

  To accurately model auctions for a particular query phrase, we model the query traffic features associated with these auctions. Due to ease of exposition, we focus on queries that contain only a single auction key-phrase[3]. For example, for a query "New York Hotels", only advertisers who bid for "New York Hotels" participate in the auctions.

- **Competitor's score**: We model the scores of "typical" competitors to the given listing L. Note that for predicting the number of impressions, we need to predict whether or not an advertisement will "win" a given auction, i.e., predict the minimum score required to win a given auction. Hence, we model the **minimum score** required to win an auction. The minimum score takes into account various factors such as bids and pclicks of the competitors, the number of winners, as well as the reserve score and other filtering criteria.

As mentioned above, to determine if the given listing L wins a generated auction, we also need to generate the score of L. We would like to stress that we **do not** model pclick. Instead, we compute the score by multiplying the provided bid value and the pclick value estimated using the pclick estimation module in production. However, typically, pclick estimation modules evolve with time. That is, at different time instances, pclick of a listing in an auction for the same query feature set can vary greatly. Hence, we use the pclick values from the latest version of the production modules as a proxy for the real run-time values. Recall that we are not trying to solve the problem of estimating pclick accurately; rather, our focus is on forecasting how a listing will fare under the existing sponsored search engine.

Next, we describe the generative model we use to model auctions in which the given listing L participates. To this end, we use a Bayes net model where each node of Bayes net corresponds to either a query traffic feature or the **minimum score** required to win the auction.

Bayes net is a popular method for modeling a set of correlated random variables or features [6]. Bayes net is represented by a directed acyclic graph (DAG) where every node of the DAG is a random variable or a query feature in our system. Edges between different nodes capture conditional dependencies between different nodes. Specifically, it satisfies the Markovian property that given its parents, a node is independent of all the other nodes that are not its descendants in the Bayes net. That is,

$$Pr(v_i|F_i) = Pr(v_i|G_i),$$

where $F_i = \{v_j | v_i \text{ is a child of } v_j\}$ and
$G_i = \{v_k | v_k \text{ is not a descendant of } v_i\}$. Now, any member of the joint distribution can be computed using the above property. Hence, to train a Bayes net, we just need to estimate $Pr(v_i|F_i)$ at each node v_i.

Why Bayes net?: Bayes net is particularly effective for categorical data as the discrete conditional probability tables (CPT) at each node can then be easy formed and manipulated to draw samples or inferences. As all of our query traffic features are categorical, and the **minimum score** can also be discretized easily, we select Bayes net to model auctions. Also, training Bayes net on categorical data is a counting process that can be efficiently implemented on a Map-reduce framework. Further, by restricting the form that joint probability distribution can take, Bayes net helps avoid overfitting to the training data. This is particularly useful for modeling auctions generated from "tail" queries, i.e., queries with a small number of searches.

[2]most of the existing ad impression forecasting tools also forecast for a time period of 1 week

[3]We use the terms key-phrase and keyword interchangeably.

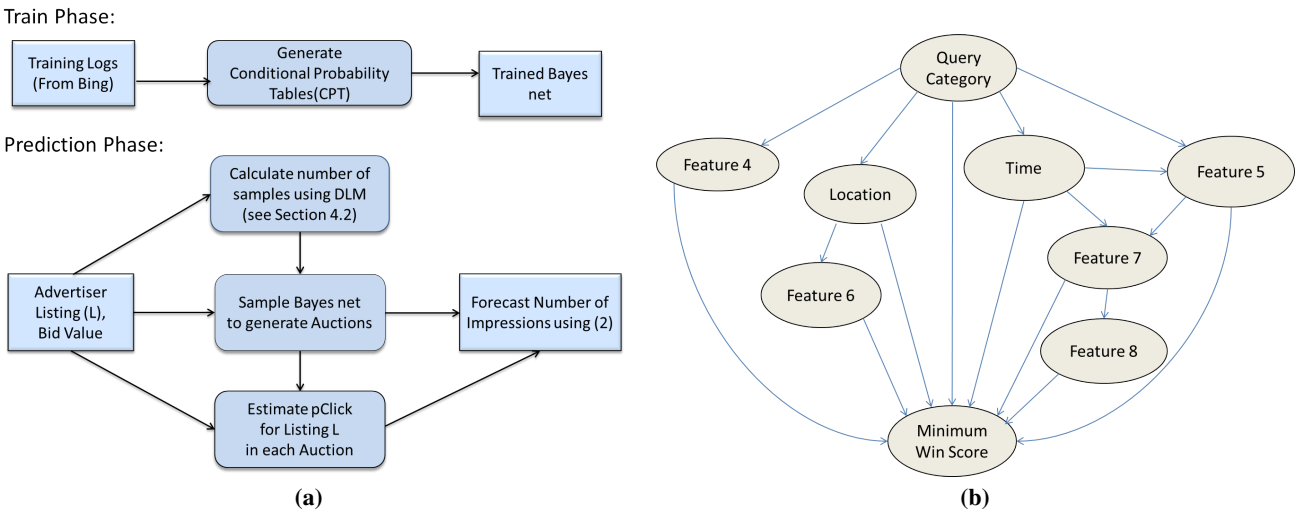


Figure 2: a) Overview of the training as well as prediction phase of our Generative Model based Ad Impression Forecasting Method (GMIF). In training phase, GMIF uses training logs to generate conditional probability tables (CPT) for each node in the Bayes net. During prediction, given a listing L and its bid value, GMIF samples the learned Bayes net to generate auctions. Number of samples required is determined by the DLM module. pclick of L for each generated auction is estimated using latest production module. Using the provided bid value and estimated pclick, GMIF computes score of L in each auction and forecasts number of impressions using (2). b) A sample Bayes net to model Auctions in Sponsored Search

Figure 2 (b) shows a sample Bayes net that we use. We do not reveal query traffic features due to confidentiality reasons. The **Minimum score** of the auction is a child node of each query traffic feature node, implying that the minimum score directly depends on all the traffic features.

As mentioned above, for training Bayes net, we estimate CPT for each node using training logs. We train one Bayes net per keyword. As the number of keywords and volume of auctions is large in a real-life sponsored search system, we use MapReduce framework to efficiently compute CPTs (see Section 4.2).

Next, for forecasting, given a listing L, we use Bayes net trained for its bidded keyword to generate sample auctions. The number of auctions to be sampled is estimated using a Dynamic Linear Model (DLM) based method that we discuss later in Section 4.1.1. After generating the sample auctions, we put listing L in each auction and estimate its pclick value for that auction using the latest production module of pclick generator. Next, using the provided bid value bid(L) we compute Score(L, Q) in each auction and compute the total number of impressions by comparing against minimum win score for the auction.

Formally, let M_L be the estimated number of auctions to be generated for the listing L. Let the set of auctions be: $\mathcal{A} = \{A_1, A_2, \ldots, A_{M_L}\}$, where $A_i = \{Q_i, MS_i\}$, Q_i consists of the query features for the auction A_i and MS_i is the minimum score needed to win A_i. Let the estimated pclick of L in A_i be denoted by pclick$(L, Q_i), \forall 1 \leq i \leq M_L$. Then, the total number of impressions is given by:

$$\text{Impressions(L)} = |\{i \mid \text{bid}(L) \times \text{pclick}(L, Q_i) \geq MS_i\}|. \quad (2)$$

See Figure 2 (a) for an overview of training and prediction phases.

Note that the predicted number of impressions critically depends on M_L, the estimated number of auctions that L will participate in test week. Estimation of M_L is challenging due to two key issues: 1) The number of searches for a given query can vary heavily across different time periods. This effect is especially prominent amongst "tail" queries that are popular for a short amount of time. For example, queries for a movie name peaks during the week of release and then drops significantly in subsequent weeks. Seasonal trends can also affect volumes of searches for a query, 2) A listing L may not participate in each auction for its bidded keyword due to several reasons. For example, an advertiser might like to target specific segment of users. Also, the sponsored search engine might filter out an ad due to its low relevance for a specific user query or to introduce randomization to do explore-exploit.

4.1 Estimation of The Number of Auctions

In this section, we describe our method for estimation of the number of auctions in which the given listing L is likely to participate in the test week. As mentioned above, prediction of the number of auctions in which L is likely to participate is challenging due to the variability in query traffic and several other factors such as exhaustion of budget, filtering by search engine etc.

To handle these challenges, we decouple the problem into two problems: 1) determine the number of searches in the test week for L's bidded keyword, 2) determine the participation ratio for L, i.e., the fraction of auctions in which L is likely to participate.

For the first problem, our approach is to model the volume of searches for a keyword as a time series and use the first order Dynamic Linear Model (DLM)—a well-known forecasting algorithm—to forecast the next point in the time series. We present details in the next subsection. For the second problem, we estimate participation ratio for each individual listing L using training logs. We present details in Section 4.1.2.

After solving these two problems, i.e., after estimation of the total number of searches and the participation ratio of L, we obtain the number of auctions M_L that L is likely to participate in in the test week using: $M_L = N \times \gamma_L$, where N is the estimated number of searches in the test week for L's bidded keyword, and γ is the estimated participation ratio of L.

4.1.1 Dynamic Linear Model (DLM)

In this section, we describe our method for estimating the number of searches a keyword is likely to obtain in the test week. As mentioned in the previous section, we use the first order DLM over the time series of the number of searches in each week.

Formally, to form the time series, we divide the time axis into bins, each of size one week. Then using logs, we compute the number of searches for each keyword in each week. Let N_t denote the number of searches in the t-th week for the bidded keyword corresponding to listing L. Also, let $1 \leq t \leq T$ where T is the test week for which we want to forecast the number of auctions.

We train the DLM using $\{N_1, \ldots, N_t, \ldots N_{T-1}\}$ and predict the number of searches in the T-th (test) week. Below, we briefly describe the first order DLM based method for time series forecasting; see [11] for more details.

The use of the first order DLMs for prediction of keyword traffic is motivated by the observation that traffic patterns are short lived and DLMs are especially well-suited for such short horizon forecasts. Assuming first order DLM, the number of searches of a particular keyword at time t is given by:

$$\begin{aligned} N_t &= \mu_t + \nu_t, \nu_t \sim \mathcal{N}(0, V), \\ \mu_t &= \mu_{t-1} + \omega_t, \omega_t \sim \mathcal{N}(0, W), \end{aligned} \tag{3}$$

where μ_t is the internal "state" of the series, $V, W > 0$ are constants, and $\mathcal{N}(0, V)$ is the Gaussian distribution with mean 0 and variance V. We assume that $\mu_0 \sim \mathcal{N}(0, C_0)$, where $C_0 > 0$ is a constant.

Now, using the above mentioned model, the following update equations can be easily derived:

$$\begin{aligned} &(N_t | N_1, \ldots, N_{t-1}) \sim \mathcal{N}(m_{t-1}, C_{t-1} + V + W), \\ &m_t = m_{t-1} + \frac{C_{t-1} + W}{C_{t-1} + W + V}(N_{t-1} - m_{t-1}), \\ &C_t = \frac{(C_{t-1} + W)V}{C_{t-1} + W + V}, \end{aligned} \tag{4}$$

where $m_0 = 0$. N_t is a random variable that corresponds to the number of searches of a given keyword; we abuse notation and denote the t-th observed value also as N_t.

For prediction, we sample $(N_t | N_1, \ldots, N_{t-1})$ using (4), while m_t, C_t are then updated using the observed value for N_t. We fix up the parameters to $W = 20$, $V = 50$, $C_0 = 100$; these values are selected using cross-validation over four weeks of data.

4.1.2 Estimation of Participation Ratio

In this section, we describe our method for estimating the participation ratio γ_L of a given listing L which is defined as the ratio of the number of auctions in which L participates in the T-th (test) week to the total number of auctions for L's bidded keyword.

Note that, in real-life systems, a listing typically does not participate in all the auctions (of its bidded keyword) due to several reasons such as budget constraints, advertiser specified targeting constraints, filtering by the sponsored search system etc. For example, if the budget of a listing is finished then it cannot participate in the future auctions for the relevant keyword. Similarly, advertisers can provide certain constraints so as to target a particular group of users only. Consequently, in practice, the participation ratio tends to be very small for several listings. Hence, estimating participation ratio γ_L is a crucial component for our method.

Note that, similar to the previous section, we can try to estimate γ_L, the participation ratio of a listing L, using time series forecasting methods. However, time span of most of the listings is a couple of weeks and hence we cannot train the DLM accurately for this problem.

To handle this problem, we make a simplifying assumption that γ_L remains constant over time; we verify the assumption over multiple weeks of real-life data. Using this assumption, we estimate γ_L by using training logs. That is, we compute the total number of wins for L in the training week and divide it by the total number of auctions that L participates in training week.

Note that the above mentioned method to compute participation ratio of L applies to the existing listings only, i.e, listings present in the logs. This poses a problem for new listings: listings that were not present in the logs in the training week. New listings themselves can be further categorized into: a) new listing by an existing advertiser, b) new listing by a *new* advertiser. Note that for the later category, no information is available to estimate participation ratio. Hence, for these listings we use a constant participation ratio, obtained by cross-validation. However, for existing advertisers, γ_L is set to be mean of the participation ratio existing listings of the same advertiser campaign. That is,

$$\gamma_L = \frac{\sum_{L_A \in \mathcal{L}_A} \gamma_{L_A}}{|\mathcal{L}_A|},$$

where $\mathcal{L}_A$ is the set of all listings in the given campaign by advertiser A which also contains listing L. Now, note that rather than computing γ_L using the above equation, we can use a constant value for γ_L. The later extension to our method, that uses constant γ_L, is referred to as GMIF-Const. For clarity, we call our former method (that uses participation ratios of other listings from the same advertiser) as GMIF-Adv. For a given advertiser campaign, typically targeting and budget allocation are same across all listings. Hence, GMIF-Adv is able to exploit information from other listings from the same campaign. Our empirical results confirm this observation as ad impression forecasts by GMIF-Adv are significantly more accurate than GMIF-Const, which ignores information from other listings of the same advertiser (see Section 5.4.2).

4.2 Large-scale Deployment

In this section, we discuss some of the issues that arise while implementing our methodology in a real-life large scale sponsored search system.

Recall that, our method proceed in two phases: 1) training, 2) prediction. For training, we generate conditional probability distribution for each edge of our Bayes net. Then, for prediction, given a listing, we sample the Bayes net to generate auctions using which we estimate the number of impressions.

Now, while training is offline, it is computationally expensive as cardinalities of some of our features is large leading to large conditional probability distribution tables (CPT). To scale to real-life sponsored search systems, we use MapReduce framework to estimate CPT from raw logs. Specifically, suppose we want to estimate the following CPT: $P(v_i | F_i)$ where $F_i = \{v_j | v_i \text{ is a child of } v_j\}$. First we find out all the unique values each $j \in F_i$ can take. Then, we *Reduce* on each combination of unique values of nodes in F_i and find the probability distribution of v_i using the obtained records. Such a scheme can be implemented easily in any standard MapReduce framework. We then use generated CPTs to construct conditional cumulative distribution functions (CDF) of each node variable. Obtained CDFs simplify and speed up the sampling process.

Next, we consider the the prediction phase of our method. Note that, this step is online, and hence requires real-time response. Consequently, we cannot sample Bayes net online to generate number of impressions. Instead, we pre-compute number of impressions

Table 1: Averaged data statistics for three train-test week combinations.

Statistics	Train	Test	Common
No. of Keywords	38991	37959	33428
No. of Listings	365187	350400	248030
No. of Auctions	6638205	6265138	Not Applicable

at all possible bid values (at small increments). Hence, when an advertiser asks for a forecast, we perform simple look-up to return predicted number of impressions at the supplied bid value. Note that if a listing is not present in the logs, but the advertiser is present. Then, we can pre-compute number of impressions for the *advertiser* at all possible bid values and for all possible keywords. If the advertiser is also, not present, then we simply store pre-computed number of impressions for each *keyword*.

5. EXPERIMENTS

In this section, we present results from large scale experiments conducted over traffic logs obtained from Bing to evaluate our framework. The goal of this section is three-fold: 1) establish that our method outperforms existing methods on several accuracy criteria, 2) demonstrate increase in coverage (% of listings for which prediction is available) over a naive baseline method that replays the training logs, 3) demonstrate scalability of our method on real-life data.

5.1 Experimental Setup and Data Statistics

In this section, we present our experimental setup and some key statistics from the data.

For conducting experiments, we select one week as the unit of time. Our initial experiments showed that the latest weekly data provides more accurate information about trends and patterns in query traffic as well as about advertiser participation as opposed to data from longer or shorter periods. Thus using weekly data leads to best accuracies for all the methods.

For our experiments, we take a randomly sample around $40,000$ search queries and select the auctions corresponding to those keywords from the training week logs (around 6.6 million). We select queries with at least 30 impressions in the training week, so that the Bayes net model can be trained with reasonable confidence. We also restrict our focus only on the listings that requires exact match between the bidded keyword and the query.

We call listings appearing in both training and test weeks *existing-listings*, while listings appearing in the test week but were not present in the training week are termed *new-listings*. Table 1 presents a few basic data statistics averaged over three train and test week combinations.

5.2 Implementation Details

In this section, we provide implementation details of our method as well as the existing methods against which we evaluate our method.

As shown in the Table 1, the data that we consider for our experiments is large scale and cannot possibly be processed using stand alone machines. Hence, to evaluate our method, we implemented a production grade prototype of our method as well as existing methods on a proprietary MapReduce platform. We use a total of nine traffic/query related features to construct the Bayes net. While the number of features considered is reasonably small, the cardinality of features tends to be very large with values up to 100,000s.

We generate the conditional probability tables (CPT) using MapReduce framework as explained in Section 4.2. We also use grid to sample the learned Bayes net for testing our method, which we call *Generative Model based Impression Forecasting (GMIF)*.

We evaluate our method against two existing methods: Training Week Replay (TWR), Normalized Bid Model (NBM) [8]. We implemented both the methods on production grid using a proprietary MapReduce platform.

TWR is a baseline method, and for a given listing L with a given bid value, it simply replays relevant auctions from training week logs. That is, it extracts out all the auctions that L participated in training week logs and compute new score for the given listing L using $\text{Score}(L, Q) = \text{pclick}(L, Q) \times \text{Newbid}(L)$, where $\text{Newbid}(L)$ is the new bid provided for listing L. TWR then simulates the auctions with new scores, computes total number of winning auctions, and forecasts it as the number of impressions for L. Note that, as TWR replays exact logs from training weeks, the participation and pclick information is not available for new-listings. Hence, this method cannot predict for new listings.

Another method that we use to evaluate our method is an adaptation of the Normalized Bid Model (NBM) by [8]. While there exist a few other approaches in literature, e.g. [1], [3], we select NBM for evaluation as it performs better or similar to the other approaches; see [8] and [3] for comparisons of these approaches. Additionally, NBM is a scalable and easy to implement approach. NBM assumes that competitor listings' "normalized scores" (defined below) are i.i.d. across listings as well as auctions.

$$\text{Normalized Score}(L') = \text{Score}(L')/\text{pclick}(L').$$

In our implementation, we sample this distribution to generate scores for competitors while generating pclick using the prediction module in production.

5.3 Evaluation Metrics

In this section, we describe various evaluation metrics we use to compare different methods. For reporting results, similar to [8], we first bin the listings according to the actual number of impressions obtained in the test week. We form four bins in all and name them as *Bin 1*, *Bin 2*, *Bin 3*, *Bin 4*. The bins are ordered in increasing order of number of impressions, *Bin 1* representing the lowest volumes, *Bin 2* representing the next highest, and so on. We do not disclose exact ranges of these bins due to confidentiality reasons.

We adopt the following metrics to evaluate performance of each method:

Relative Error (RE): Relative error of a listing is given by: $RE = \frac{|predicted - actual|}{actual}$, where $predicted$ is the predicted number of impressions for the listing and $actual$ is the actual number of impressions in the test period. We report average relative error over all the test weeks listings. This is a standard metric in forecasting and was used by [8] and others.

Accuracy: We measure the accuracy of a method for a bin as the number of listings (in the bin considered) that have less than $\tau > 0$ relative error. That is,

$$\text{Accuracy}_\tau = \frac{|\text{Listings s.t. RE } \leq \tau|}{\text{Number of Listings}}. \quad (5)$$

τ is a threshold parameter, where smaller values of τ imply more stringent accuracy measures. We report results for different τ values; default value of τ is 0.5.

Accuracy measures percentage of listings for which the prediction is within a factor τ of the actual prediction. That is, it accounts for the volume of "reasonably good quality forecasts. Note that while using this measure, the very low impression bins will generally show poorer results. This is because of the low values of denominator. For example, assume a listing actually gets 2 im-

pressions while the prediction was 4 for a specific bid value; the RE value for this item will be $+100\%$. The RE number in this case can be misleading since the absolute values are not greatly different. Higher impression bins do not face this problem and give results which are more accurately indicative of the true quality and are important because they belong to bigger advertisers with more budget to spend on their campaigns. Thus while using this measure, though we will provide numbers for both the high and low impression bins, we will specially focus more on the accuracy numbers for the higher impression bins to gain fair insight into the quality of our predictions.

We also report overall accuracy of methods across bins as well as bin-wise averaged accuracies, i.e., average of accuracy obtained in each bin.

F-measure: We report the F-measure to compensate for the deficiencies of the Accuracy metric in lower impression bins. Given the pre-defined bins on actual impressions, we calculate the true positives (predictions in the specific bin and actuals in the same bin too: tp), false positives (predictions fall in the bin but actuals lie in some other bin: fp) and false negatives (predictions made outside of the bin but actuals fall in the specific bin: fn) for each of the bins. The precision and recall for each bin is then calculated as $Pr = \frac{tp}{tp+fp}$ and $Re = \frac{tp}{tp+fn}$ respectively. We also calculate the F-measure which is the harmonic mean of precision and recall: $F = \frac{2Pr.Re}{Pr+Re}$. F-measure is a well known performance measure and is widely used in a number of domains like information retrieval [7].

We also report the additional number of new listings we can predict which did not appear in training week. New listings can be further categorized into: 1) both the advertiser and the keyword were present in the training data but the listing is new to the system, 2) only the corresponding keyword was available in the training data and the advertiser (and so also the listing) is new to the system. We report accuracy and other metrics for both the categories and show that our methods can effectively exploit information available from the advertiser's other listings.

Note that we can measure metrics mentioned in this section for only those listings which do not change their bids in the entire test week window. If a listing changes bid during test week, then it is not clear which bid value to select for forecasting impressions. However, we observe that approximately 80% of the listings do not change bids, hence we can report results over a substantial fraction of listings.

5.4 Results

In this section, we report results of several experiments with multiple weeks of data for evaluating forecasts from our method against existing methods. We use evaluation metrics described in the previous section to compare different methods. Recall that, we name our method as Generative Model based Impression Forecasting (GMIF), a baseline method as Test Week Replay (TWR), and the method by [8] as Normalized Bid Model (NBM). Also, recall that, for reporting results, we bin the listings into four bins: *Bin 1*, *Bin 2*, *Bin 3*, *Bin 4*.

5.4.1 Existing Listings

Here, we report results regarding forecast of impressions for existing listings, i.e., the listings that are present in both training and test week logs. For our experiments, we obtain data from logs of a recent month that shows considerable fluctuations in traffic volume of keywords and distribution of traffic parameters due to reasons such as shopping season surges. As the traffic volume and patterns change week by week, the results clearly bring out the advantages of our method (generative modeling and short horizon time trending) compared to the others.

First, we present average accuracies (see Section 5.1) obtained by all the three methods. Figure 3 compares the average accuracies of the three methods in each of the four bins for $\tau = 0.3, 0.5, 0.6$. The bins are arranged on the axis in order of increasing test week actual impressions. As mentioned earlier, the boundaries of the bins could not be revealed due to confidentiality reasons. Figure 3 also reports the accuracy variances across the multiple test weeks considered. Clearly, our method GMIF outperforms both TWR and NBM in terms of average accuracy for all bins for all threshold values except for Bin 1 with $\tau = 0.3$. GMIF also consistently achieves the lowest variance. NBM approach outperforms TWR in all bins except for the lowest one.

Hence, our method is more accurate as well as more consistent than the existing methods. We believe, low variance is due to our more principled generative model based approach to modeling auctions in comparison to other methods that are prone to overfitting to the training data. Note that although NBM models competitor scores by a simplifying assumption, it assumes exactly the same number of auctions as the past week and uses exactly the same query traffic.

Next, in Table 2, we report the overall accuracy across all the bins (weighted by the number of listings) as well as the accuracy averaged over all the bins (unweighted averaging). Clearly, our method achieves significantly higher accuracy than both TWR and NBM. In term of overall accuracy, GMIF is approximately 7% and 19% more accurate than TWR and NBR, respectively. Similarly, in terms of bin-wise accuracy, GMIF is around 12% and 15% more accurate than TWR and NBM, respectively. Bin-wise accuracy is a widely used metric, but it is particularly useful for the sponsored search scenario as it takes into account the effect of heavy tail distribution of impressions (i.e., many impressions in Bin 1) as well, which is typical to sponsored search.

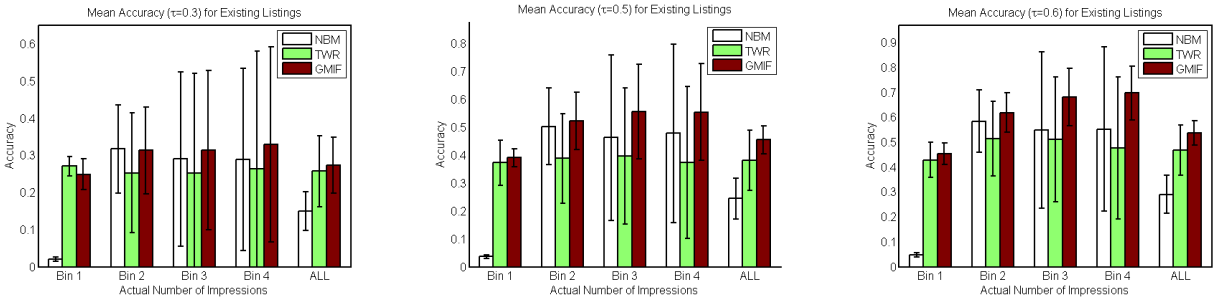
Method	Overall Accuracy	Bin-wise Accuracy
NBM	0.24	0.35
TWR	0.38	0.38
GMIF	0.45	0.50

Table 2: Overall and bin-wise accuracy ($\tau = 0.5$) for different methods. Our method (GMIF) is at least 7% more accurate (according to overall accuracy) than NBM and TWR.

Now, observe that in Figure 3, the accuracy in the lowest impression bucket (Bin 1) is poor for all the methods. Reason being, in Bin 1, due to small number of actual impressions, the denominator (actual) values in calculating RE are small that leads to large RE. As discussed earlier, this is an inherent drawback of any relative error based metric. To alleviate this problem, we also report F-measure obtained within each bin by the three methods considered. Figure 4 (a) reports the average F-measures for the three methods and also show variance in F-measure for each of the method. Here again, our method (GMIF) consistently outperforms the other methods in all the bins. However, we note that for the F-measure metric, TWR outperforms NBM in all the bins consistently and is closer to GMIF across all the bins.

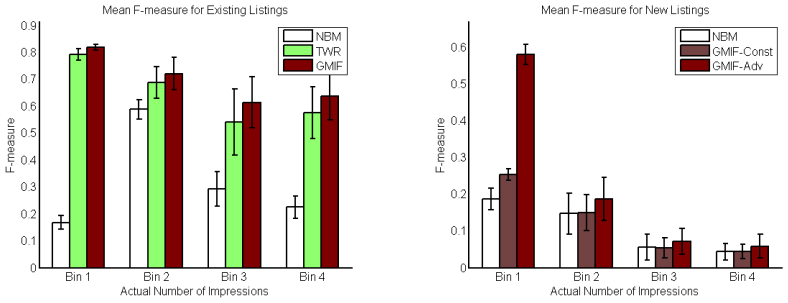
5.4.2 New Listings

Here, we compare different methods for their predictions on new listings, listings that are present in the test week logs but not in the training week logs. We did similar set of experiments on new listings as on the existing ones and used the same metrics to com-



(a) (b) (c)

Figure 3: Average accuracy with different thresholds for existing listings: a) $\tau = 0.3$, b) $\tau = 0.5$, c) $\tau = 0.6$. X-axis represents actual number of impressions in test data, and is divided in four bins. Bin "ALL" represents all the listings, irrespective of the number of impressions. Our method (GMIF) consistently obtains higher accuracy than both NBM and TWR, while obtaining smallest variance of the three methods across all bins and all threshold τ values.



(a) (b)

Method	Overall Accuracy	Bin-wise Accuracy
NBM	0.05	0.19
GMIF-Const	0.05	0.21
GMIF-Adv	0.15	0.24

(c)

Figure 4: a) Average F-measure for Existing Listings. Our method obtains around 15% higher F-measure than TWR and about 25% higher than NBM, in Bin 2 which forms the torso of the impression range and is one of the most interesting bins for real systems. b) Average F-measure for New Listings. GMIF-Adv obtains 32% higher F-measure than GMIF-Const and 31% higher F-measure than NBM.c) Overall and bin-wise accuracy ($\tau = 0.5$) for different methods for new listings.

pare different approaches. Since the Training Week Replay (TWR) method uses logs to determine pclick as well as participation information, we cannot use it to predict for new listings. However, we can use our adaptation of the NBM method of [8] in this case.

Recall that new listings can be further divided into: 1) listings whose advertiser was present in the past week, 2) listings with new advertisers. As explained in Section 4.1.2, our method can be adapted to both of these categories. We refer to our extension of GMIF method that uses advertiser information as GMIF-Adv, while the GMIF method adaptation that does not use advertiser information is referred to as GMIF-Const.

Figure 5 shows the average accuracy in each bin obtained by NBM as well as our GMIF-Const and GMIF-Adv method for $\tau = 0.3, 0.5, 0.6$. Note that while GMIF-Const and GMIF-Adv perform essentially equally well for the higher impression bins, GMIF-Adv does distinctly better in the lowest bucket which indicates that advertiser specific information plays a significant role in determining the number of impressions for listings with small impression volume (tail listings). The figure also shows that both the GMIF approaches almost always give higher average accuracies and lower variances compared to the NBM approach.

Similar to the previous section, we also report overall accuracies and bin-wise mean accuracies for NBF and our GMIF methods (see Figure 4 (c)). Here again, GMIF-Adv outperforms the other methods in both the measures. The gain in overall accuracy is substantially higher compared to the gain in bin-wise accuracy measure.

Figure 4 (b) shows the corresponding F-measures for the three approaches in different bins. Note that while considering this metric, GMIF-Adv performs consistently better compared to NBM and GMIF-Const which perform almost similar for the higher impression bins.

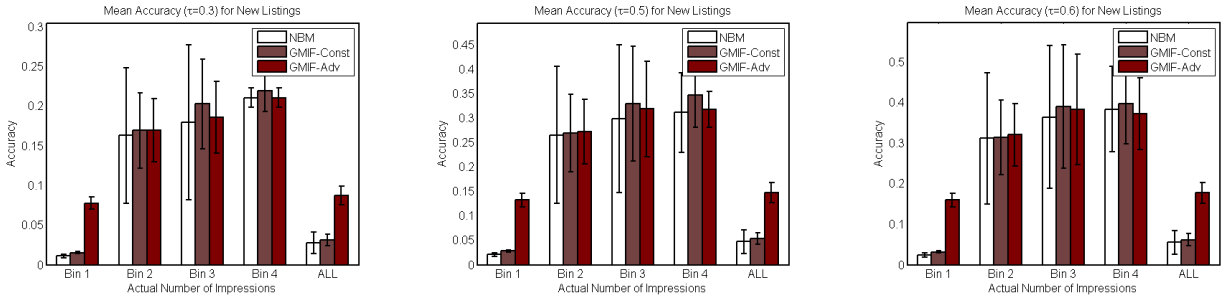
Table 3: Decrease in average Relative Error w.r.t. Baseline NBM method.

Method	Bin 1	Bin 2	Bin 3	Bin 4	ALL
GMIF-Const	5.20	11.99	16.54	16.47	5.66
GMIF-Adv	36.38	23.03	18.63	22.04	36.49

For new listings we also report gain in relative error for both the GMIF approaches compared to NBM in each bin (see Table 3). This metric clearly, shows that our methods are significantly better than NBM in each bin and overall. It also corroborates the fact that GMIF-Adv achieves maximum relative gain in the lower bins where we observed that the influence of advertiser campaign information is a crucial factor in predicting impressions (see Section 4.1.2).

5.4.3 Dynamic Linear Model

In this subsection we take a deeper look at the accuracy of our traffic prediction module that is based on first order dynamic linear models. Note that the accuracy of this module in the entire architecture needs special attention since this predicts the number of sample auctions in which a listing will participate in the test period. Inaccuracy in this step should lead to a ripple effect on the error as it magnifies inaccuracies of the overall impression forecasts. As mentioned in Section 4.1.1, we learn a DLM for each keyword using past 6 week traffic volumes as time series points. We then tune the DLM parameters for optimality using cross validation over mul-



(a) **(b)** **(c)**

Figure 5: Average accuracy with different thresholds for new listings: a) $\tau = 0.3$, b) $\tau = 0.5$, c) $\tau = 0.6$. GMIF-Const and GMIF-Adv are adaptations of our GMIF method to new listings, with main difference being that GMIF-Adv uses existing advertiser's information, while GMIF-Const ignores that information. Clearly, GMIF-Adv is significantly better than NBM and in Bin 1, outperforms GMIF-Const as well.

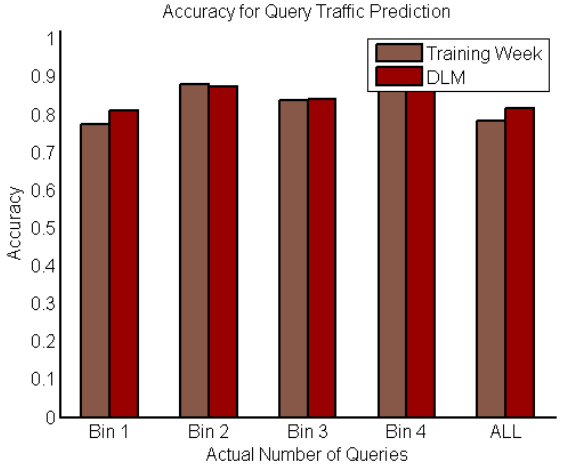


Figure 6: Average accuracy in traffic volume prediction (with $\tau = 0.3$). First order DLM based method achieves significantly higher accuracy than baseline method (using training week traffic volume as approximation to test week traffic volume), especially for Bin1 which constitutes 80% of the traffic.

tiple weeks of learning and use the trained and tuned models for prediction of traffic volumes of keywords. We report relative error/accuracy (defined in Section 5.3) to evaluate gains from using the DLM framework. We compare against a naive but a very effective method in practice: given a keyword assume its last week's traffic volume as the estimate for the next week's traffic volume. Note that both TWR and NBM methods obtain their number of samples using this baseline heuristic. Figure 6 reports the accuracy for the baseline (training week volume) and DLM prediction method with $\tau = 0.3$. We observe similar results for other values of τ. The Bins on the horizontal axis are indicative of real traffic ranges of the test week and are not disclosed for confidentiality reasons. The Bins are arranged in increasing order of test week actual traffic volume from Bin1 to Bin4. It can be observed from Figure 6 that DLM predictions overall show better results compared to the baseline method. Bin1 is of special interest as it represents torso and tail queries and contain around 80% of the total number of keywords. For this bin, our DLM based method is significantly better than baseline method (by 4%). These results show that first order DLMs can learn traffic volume trends reasonably well from only a few points in time series data. This is especially needed in our specific case since most torso and tail keywords have short lifecycles of existence (typically ranging from 6-8 weeks).

5.4.4 Run-times

Finally, in this section we report approximate time required by each step of our approach on a standard proprietary grid platform. For a substantial fraction of production data from Bing, our entire pipeline in offline mode takes approximately 260 mins to run end to end. We provide detailed average time breakups in Table 4

Table 4: Approximate Processing Time

Step	Processing time
Traffic Feature Extraction	192 mins
Train Bayes net	19 mins
DLM based Traffic Volume Prediction	10 mins
Sampling Auctions from Bayes net	12 mins
Simulation on artificial traffic	27 mins

The first row in the above table reports time required to extract query traffic features from raw logs. This time step is most expensive and takes about three hours. Next row shows time required for training the Bayes net. Fourth and fifth row shows the total time required to forecast number of impressions for all the advertisers, which is approximately one hour only. Hence, our method's training as well as pre-computation needed for real-time prediction finishes within only half a day.

6. DISCUSSION AND CONCLUSION

In this paper, we considered the problem of ad impression forecasting, which is critical in helping advertisers optimize their return on invest from sponsored search advertising.

Most of the existing methods view the problem from a game-theoretic point of view, where the goal is to model how competitor's for an ad are going to bid. But they mostly ignore or overfit to the query traffic information which, via pclick, also has a significant impact on the chances of an ad winning an auction. In this paper, we modeled the auctions holistically using a carefully designed Bayes net that captures explicitly the correlation between competitors' scores and query traffic features. Our empirical results corroborate our view that the interplay between the query traffic features and competitors' scores plays a significant role and needs to be captured using a detailed model.

While in this paper we focused on predicting the number of impressions, our method is flexible and allows for various extensions. For example, we can extend our method to estimate other performance indicators such as number of clicks, average position of an ad. Further, our current method is designed assuming that only ad-

vertisers who bid for the exact user query can participate in its auctions. However, in real-life systems, a listing can participate in the "related" keywords' auctions as well. For example, a listing that bids on "shoes" can participate in a user query "running shoes". We can extend our method to such cases by forming a mapping between the bidded keywords and user queries for an advertiser by using logs. In future, we plan to conduct rigorous experiments using our method's extension for such listings. Finally, we used a first order DLM to capture trends and momentary peaks in query search volume. However, the DLM ignores information about query traffic features and only focuses on the total number of searches. We plan to address this limitation using dynamic Bayes net, that smoothly vary conditional probability distributions over each edge. Bayes net also allows for feature targeting: For example, if the advertiser has a certain geographical area, or a demographic group to which they would like to advertise, then by fixing the corresponding nodes in the Bayes net, we can generate the traffic corresponding to the targeting.

Acknowledgements

We would like to thank A. Kumaran, Krishna Leela Poola, B. Ashok, and Sayan Pathak for several discussions that helped in shaping core ideas of the paper and also for detailed comments on an initial draft of the paper.

7. REFERENCES

[1] S. Athey and D. Nekipelov. A structural model of sponsored search advertising auctions. In *Technical report, Microsoft Research*, May 2010.

[2] Y. Cui, R. Zhang, W. Li, and J. Mao. Bid landscape forecasting in online ad exchange marketplace. In *KDD*, pages 265–273, 2011.

[3] Q. Duaong and S. Lahaie. Discrete choice models of bidder behavior in sponsored search. In *WINE*, 2011.

[4] 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.

[5] T. Graepel, J. Q. Candela, T. Borchert, and R. Herbrich. Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine. In *ICML*, pages 13–20, 2010.

[6] D. Koller and N. Friedman. *Probabilistic Graphical Models*. MIT Press, 2009.

[7] C. D. Manning, P. Raghavan, and H. Schtze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[8] F. Pin and P. Key. Stochastic variability in sponsored search auctions: observations and models. In *ACM Conference on Electronic Commerce*, pages 61–70, 2011.

[9] H. R. Varian. Position auctions. *International Journal of Industrial Organization*, 25 (6):1163–1178, 2007.

[10] X. Wang, A. Z. Broder, M. Fontoura, and V. Josifovski. A search-based method for forecasting ad impression in contextual advertising. In *WWW*, pages 491–500, 2009.

[11] M. West and J. Harrison. *Bayesian Forecasting and Dynamic Models*. Springer Series in Statistics, 1997.