Learning Parametric Models for Context-Aware Query Auto-Completion via Hawkes Processes

Liangda Li[†], Hongbo Deng[‡*], Jianhui Chen[†], Yi Chang[¶*]
[†]Yahoo Research, 701 First Avenue, Sunnyvale, CA 94089
[‡]Google Inc., 1600 Amphitheatre Pkwy, Mountain View, CA 94043
[¶]Huawei Research America, 2330 Central Expy, Santa Clara, CA 95050
{liangda, jianhui}@yahoo-inc.com, hbdeng@google.com, yi.chang@huawei.com

ABSTRACT

Query auto completion (QAC) is a prominent feature in modern search engines. High quality QAC substantially improves search experiences by helping users in typing less while submitting the queries. Many studies have been proposed to improve quality and relevance of the QAC methods from different perspectives, including leveraging contexts in long term and short term query histories, investigating the temporal information for time-sensitive QAC, and analyzing user behaviors. Although these studies have shown the context, temporal, and user behavior data carry valuable information, most existing QAC approaches do not fully exploit or even completely ignore these information. We propose a novel Hawkes process based QAC algorithm, comprehensively taking into account the context, temporal, and position of the clicked recommended query completions (a type of user behavior data), for reliable query completion prediction. Our understanding of ranking query completions is consistent with the mathematical rationale of Hawke process; such a coincidence in turn validates our motivation of using Hawkes process for QAC. We also develop an efficient inference algorithm to compute the optimal solutions of the proposed QAC algorithm. The proposed method is evaluated on two real-world benchmark data in comparison with state-of-art methods, and the obtained experiments clearly demonstrate their effectiveness.

Keywords

query autocompletion;Hawkes process;contextual data

1. INTRODUCTION

Query auto completion (QAC) [3] is one of the most important features in modern search engines; it helps users in a wide range of information pursuing activities, precisely, getting desirable information quickly but with as little typing effort as humanly possible. Upon receiving a user's input entered into a search engine's search box, QAC engine instantly displays to the user a list of recommended query completions, each of which usually starts with the user's input as its prefix. Following each character subsequently entered into the search box by a user, QAC engine updates the recommended query completions with a presumptively better reflection on the user's underlying search intentions. Signaled by a click on one of the recommended query completions, the user's information need is satisfied and his interactive process with the QAC engine is ended. The most basic QAC algorithm is the approach of *mining wisdom of the crowds* by suggesting the completions that are most popular among users in the past, generally referred to as MostPopularCompletion (MPC) [2]. Although providing satisfactory query completion candidates on average, MPC's prediction quality is far from optimal to satisfy different information needs from different users. Suppose a user is submitting a query and has already typed *"sig"* into the search box. Based on MPC, the top-ranked query completions include *"signature"* and *"sigma"*, as both queries have very high frequency of occurrence. However, if the user has recently submitted *"acm conferences"* as a query, some other query completion candidates, for example, *"sigkdd"*, *"sigir"* and *"sigmod"*, could better reflect this user's real search intent. This illustrative example demonstrates that simply relying on the popularity of the queries does not guarantee reliable QAC prediction performance. Intuitivelly it makes more sense to leverage certain contextual and other relevant information for a potential boosting in the quality of query completion prediction.

Many studies have been proposed to improve quality and relevance of the QAC methods from different perspectives, including leveraging contexts in long term and short term query histories [2], investigating the temporal information for time-sensitive QAC [32, 34], learning to combine more personalized signals [31], analyzing user behaviors [21, 37, 18], etc. Although certain progress has been made in previous studies, most of them focus on recommending query completion candidates according to the similarity scores between the candidates and the rich representations of the contextual data. Moreover, these existing methods usually build predictive models which use only a single type of contextual information, for example, time sensitive model, or query history sensitive model. Apparently these type of methods could not attain the best prediction performance, as they do not fully exploit the valuable yet imperceptible information carried in the rich contextual, temporal, and position of the recommended query completions (position in short), which results in an ambiguous guess why a user clicks a certain suggestion among all candidates.

We consider to appropriately model the influence between users' click choices across different QAC sessions, which arise from three representative factors: context, temporal and position information. Basically, recent queries, recently visited web pages, and recent check-in locations are examples of online activities that may be

[*]Work done while at Yahoo.

WSDM 2017, February 06-10, 2017, Cambridge, United Kingdom

DOI: http://dx.doi.org/10.1145/3018661.3018698

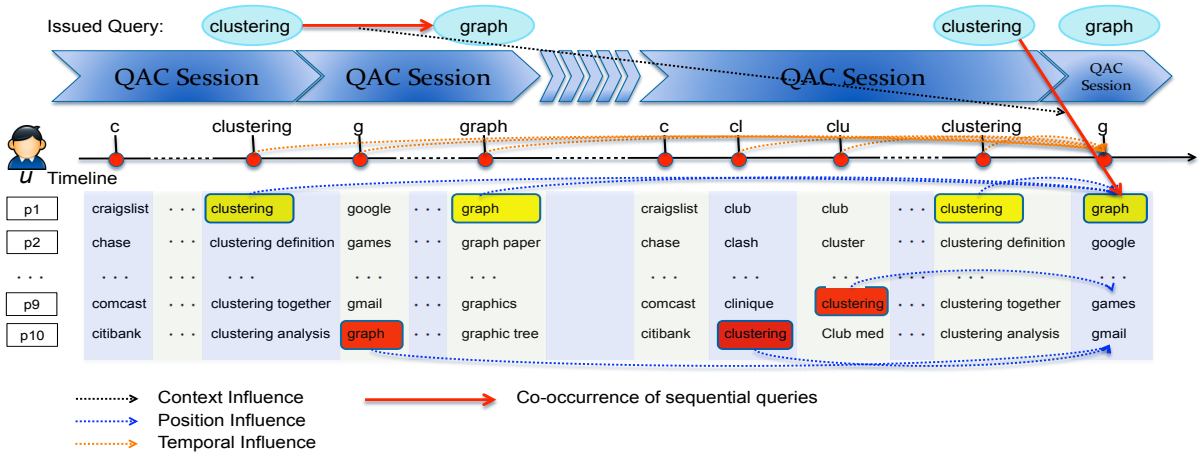


Figure 1: Illustration of how different factors contribute to the influence between Click Events in different QAC sessions. *Yellow tag highlights the query finally clicked by the user, while red tag highlights the user's intended query which however was not clicked. Black dot line represents the context influence between query pairs, orange dot line denotes the temporal influence based on timestamp, and blue dot line indicates the position influence.*

viewed as different possible user contexts [2]. In this paper, our focus is on the user's recent queries in his/her search history. As shown in Figure 1, for each issued query, there is a corresponding QAC session along with its timestamp in the timeline, where a QAC session consists of the behaviors from the first keystroke a user typed in the search box towards the final submitted query. Yellow tag highlights the query a user finally clicks at a certain position and timestamp, while red tag highlights the user's intended query he/she does not click at that position and timestamp. In Figure 1, two consecutive queries "clustering" and "graph" are issued by user u, then the previous query "clustering" can be viewed as the context of query "graph". Generally, two consecutive queries issued many times by different users are more likely to have a strong correlation between each other. It makes more sense to take into account the explicit temporal information of query sequences exhibited by many different users in the whole query logs. The basic intuition is that if two consecutive or temporally-close queries are issued many times by the same user or many other users, it is more likely these two queries are semantically related to each other. We define *query influence* as the occurrence of one query raises the probability that the other query will be issued in the near future. Such influence will be very useful for predicting the next issued query. The co-occurrence of sequential queries, e.g., "clustering" and "graph", will influence the later QAC prediction given related context, which is referred as *context influence* in this paper. In addition, *temporal influence* exists and will be used to measure the degree of the influence between two queries from the temporal aspect. Roughly speaking, temporally-close queries are more likely to have higher influence between each other. Furthermore, it is very important to model the position of the intended query shown in the suggestion list. As we can see in Figure 1, although some intended queries (highlighted with red tag) were shown in a relative lower position for short prefixes, most users tend to click them when showing in higher position. Thus, there is a type of *position influence* regardless of the query itself.

To model the influence between users' click behaviors across different QAC sessions, we resort to the Hawkes process [15]. Hawkes process is a temporal point process, which has been widely used for solving various data mining and machine learning research problems [22]. A typical Hawkes process is characterized by a stochastic (conditional) intensity function, precisely describing how likely an event of interest will happen at each time point; specifically this intensity function consists of a base intensity term as well as a self-exciting term; the former describes the background rate of this point process, independent from any past history of the event occurrences; the latter origins from the well-known *self-exciting* property of the Hawkes process, implying the existence of a positive influence of the past event on the current one, which other popular machine learning algorithms do not naturally possess. Such *self-exciting* property are expected to naturally capture the influence between the events that search engine users make clicks in each QAC session. However, existing Hawkes models are generally designed to model the influence between events based on their temporal distance. To comprehensively utilizing content, temporal, and position information in modeling the influence between click events in QAC sessions, a novel Hawkes model is desirable for integrating all above three factors in influence modeling.

In this paper we proposes a novel Hawkes model which contains two parts: 1) *the base intensity*: it implies how likely a search engine user clicks a suggestion based on its own property, such as the general popularity in the query log, how frequent this user submits the query, etc.; this type of information is generally independent from this user's previous search history. 2) *the self-exciting influence*: it is jointly measured by the context, temporal, and position difference between the historical QAC sessions and the current QAC session. Like time-decay kernels that are widely used in existing Hawkes models in capturing the self-exciting property from the temporal aspect, we design a space-decay kernel to capture the influence from the spatial aspect. Furthermore, we propose a set of contextual features based on the co-occurrence of query submissions in certain range of histories. Such features are designed to illustrate the influence between user clicks from the contextual aspect. We evaluate our method on two real-world high-resolution QAC logs collected from Yahoo on both PCs and mobile phones. We compare the performance of our model with state-of-the-art QAC algorithms. Experimental results show that the proposed method can predict queries that better satisfy users' search intent. Moreover, the learned model provides us insights into how different factors contribute to the influence between search engine users' click behaviors across different QAC sessions.

2. PROBLEM DEFINITION

Consider a typical scenario with M users, where each user issues a query sequence. For the specific m-th user, the QAC log records N_m QAC sessions; the n-th QAC session contains $S_{m,n}$

keystrokes, and each keystroke contains D suggested queries; the n-th QAC session has $S_{m,n} \times D$ potential slots, and each slot corresponds to one suggested query. Denote $z_{m,n,s,d} = 1$ if the user clicks the d-th slot from the s-th keystroke, and $z_{m,n,s,d} = 0$ otherwise. Denote by $x_{m,n,s,d}$ the relevance feature of a prefix-query pair, where the prefix is the one at the s-th keystroke and the query is from the d-slot at the s-th keystroke. Given the weights ω of those relevance features, we expect $\omega x_{m,n,s,d}$ to reveal the relevance score of each suggested query.

2.1 A High-Resolution QAC Log

Traditionally the search query logs only include the submitted queries and their associated search results; they do not log the sequential keystrokes (prefixes) entered into the search box, nor do they log the corresponding QAC suggestions. To better analyze and understand users' real behaviors, a *high-resolution QAC log* is recently introduced and discussed [23]; it records users' interactions with a QAC engine at each keystroke and associated system response in an entire QAC process. In the high-resolution QAC log, each submitted query is associated with a *QAC session*, which corresponds to the set of search results obtained from the first keystroke entered into the search box towards the final submitted query. Specifically, the recorded information in each QAC session includes each keystroke the user has entered, the timestamp of every keystroke, the corresponding top 10 suggested queries to a prefix, the anonymous user ID, and the final clicked query.

Let us use a toy example to briefly introduce how a user interacts with a QAC engine and makes the final click in an entire QAC session. As shown in Figure 1, a QAC session contains S keystrokes and each keystroke has a suggested query list of length D.[1] A QAC session ends at the keystroke where the user clicks a query in the suggested query list, or when the prefix at that keystroke is exactly the query the user enters into the search engine. Among the $S \times D$ *slot*s in each QAC session, where each slot q_{ij} is indexed by the i-th position at the j-th keystroke, a user clicks at most one of them, although the user intended query may appear in many slots. Since users' clicked queries are usually their intended queries, appropriate modeling of users' click actions can be a good solution of the QAC problem. The ideal QAC engine should be able to rank the user intended query higher with less keystrokes or short prefixes. Our algorithm aims at leveraging such a QAC log data to get a better understanding of user's sequential behavior in the QAC process.

2.2 Hawkes Process

One powerful tool in statistics for modeling event sequence data is Hawkes process, which is a class of self- or mutually-exciting point process models [15]. A univariate Hawkes process $\{N(t)\}$ is defined by its intensity function

$$\lambda^*(t) = \mu(t) + \int_{-\infty}^{t} \kappa(t-s) dN(s),$$

where $\mu > 0$ is a base intensity, κ is a kernel function capturing the positive influence of past events on the current value of the intensity process, which is the process's *self-exciting* property that the occurrence of one event in the past will trigger events happening in the future. Such self-exciting property exists either between every pair of events as assumed in a normal univariate Hawkes process, or only between limited pair of events.

In the context of query auto-completion, the event we are interested in modeling is how likely a user m will click a suggested query located at the d-th position and the s-th keystroke in a QAC session. The key idea behind this modeling is to capture the influence between those click events, i.e., what motivates a user to make a click choice, and borrow some knowledge from previous click events for better prediction quality in current time stamp. Since a user makes only one click in every QAC session, the n-th click event in the entire click event sequence of the same user is the click event occurring at the n-th QAC session.

[1] We experiment with real-world QAC logs where $D = 10$.

Table 1: Patterns in Constructing Contextual Features

Pattern p	Description
$q' \rightarrow q$	query q is submitted just after the submission of query q'.
$q \longleftrightarrow q'$	query q and q' are submitted in adjacent.
$q' \xrightarrow{(v)} q$	query q is submitted after the submission of query q', and v queries have been submitted in between.
$q \xleftrightarrow{(v)} q'$	v queries are submitted between the submission of query q and q'.

We identify three factors which play key roles in characterizing how a user makes the click choices; they are summarized below.

Slot The spacial slot information, i.e., the displayed position of the suggested query, is a factor which obviously affects the influence between click events. To quantify the degree of the influence between the click events from the spacial slot aspect, we use the following formula:

$$\kappa(|\mathbf{p}_l - \mathbf{p}|)$$

where $\mathbf{p}$ is slot where a user makes the current click, and $\mathbf{p}_l$ is the slot where the user makes l-th historical click event, i.e., the click occurs at the l-th QAC session, and $\kappa(|\mathbf{p}_l - \mathbf{p}|)$ represents a decay effect from the slot discrepancy. Notice that $\mathbf{p} = (i, j)$ is a vector of length 2; its entries i and j respectively denote the position and the keystroke.

Timestamp Analogously, the timestamp information, i.e., the temporal stamp whether the click event occurs, is another important factor. To quantify the degree of the influence between click events from the temporal aspect, we use the following formula:

$$\kappa(t_l - t)$$

where t is timestamp when a user makes the current click, t_l is timestamp when the l-th historical click event occurs, and $\kappa(t_l - t)$ represents a time decay effect.

Context As explained in previous sections, rich contextual data carries value information for the query suggestion prediction. To quantify the degree of the influence between click events from the context aspect, we design a set of contextual features that describe the relationship between the content of a historical query q' and a current suggested query q. These features count the number of appearances of a certain pattern involving both the historical query q' and the current suggestion q in a certain time range formulated as:

$$x(p)(t, \Delta t) = \#\{p \in [t - \Delta t, t)\},$$

where p represents a certain defined pattern, $[t - \Delta t, t)$ is the time interval from some ancient timestamp to the current timestamp. Table 1 shows several patterns we adopt in this paper. Our feature design is inspired by the features proposed in [27]. The novelty of our design is that we propose features in more general forms, and also explore brand-new patterns under the scenario of query auto-completion, thus produce far more features.

As shown in Table 1, our features generally originate from the co-occurrence of two queries in the query sequence submitted by search engine users, and reflect pairwise relationship. We form a feature vector $\mathbf{x}_{q',q}(t)$ for each query-pair (q', q) at any given timestamp t as

$$\mathbf{x}_{q',q}(t) = \{x(p)(t, \Delta t)|p \in \mathcal{P}_{q',q}, \Delta t > 0\},$$

where $\mathcal{P}_{q',q}$ refers to the set of patterns involving the pair of queries $\{q', q\}$. Thus for each timestamp t, a unique set of feature vectors $\{\mathbf{x}_{q',q}(t)\}$ imply how a historical click event (on query q') influence the current click event (on query q) from the context aspect.

2.3 Factorial Hawkes

Recall that we have identified three key factors which contribute to the influence between historical click events and the current click in Section 2.2. To simultaneously leveraging these factors and make them capture the actual influence exists between click events across QAC sesssion, we build a univariate Hawkes process on each user's issued query sequence inspired by [22]. Specifically consider a click by user m in the n-th QAC session issued at timestamp $t_{m,n}$ on a suggestion $q_{m,n,s,d}$ located at the slot $\mathbf{p}_{m,n} = (d, s)$, its intensity function can be mathematically expressed as

$$\lambda(t, \mathbf{p}) = \mu + \sum_{t'<t} \beta \mathbf{x}_{q',q}(t)(\kappa(t - t') + \alpha\kappa(|\mathbf{p} - \mathbf{p}'|). \quad (1)$$

The formulated intensity function in Eqn (1) consists of two components; the first component corresponds to the baseline intensity μ, which captures how often a user make a click spontaneously[2], i.e., not triggered by any other click event. Conventionally the baseline intensity μ is set as a constant, which can be computed from the historical data. This approach however may not accurately represent the value of μ. In our approach, we parametrize μ as a functional of a set of widely used features $\mathbf{x}'$ used in existing QAC methods. For similarity, given a specific feature vector $\mathbf{x}'$, we denote μ as a linear function as

$$\mu = \omega \mathbf{x}',$$

where the weight vector ω indicates the importance of the each feature entry and its value will be computed adaptively.

The second component in Eqn (1) is designed to reflect the influence between historical click events to the current click event. It systematically takes into account the spacial, temporal, and context information, where $\mathbf{x}_{q',q}(t)$, $\kappa(t - t')$[3], and $\kappa(|\mathbf{p}' - \mathbf{p}|)$, correspond to the context, temporal, and spacial information, respectively. $\beta > 0$ is a balance parameter, indicating the relative importance of the baseline intensity and the contextual information.

From the proposed Hawkes model with the intensity function in Eqn (1), given a set of observed click event sequences described by temporal data $T = \{N_m(\cdot)\} = \{\{t_{m,n}\}_{n=1}^{N_m}\}$, spacial data $P = \{p_{m,n}\}$, and query content $Q = \{q_{m,n}\}$, we can denote its likelihood as

$$\mathcal{L} = \sum_{m=1}^{M}\sum_{n=1}^{N} \log \lambda_m(t_n, \mathbf{p}_n) - \sum_{m=1}^{M}\sum_{d=1}^{D}\sum_{s=1}^{S}\int_0^T \lambda(t, \mathbf{p})dt. \quad (2)$$

[2]For simplicity, we assume this cascade-birth process is a homogeneous Poission process with $\mu(t) = \mu$.

[3]Our paper uses the exponential kernel in experiments, i.e., $\kappa(\Delta t) = \omega' e^{-\omega' \Delta t}$ if $\Delta t \geq 0$ or 0 otherwise. However, the model development and inference is independent of kernel choice and extensions to other kernels such as power-law, Rayleigh, non-parametric kernels are straightforward.

3. INFERENCE

To elucidate the maximization of the log-likelihood in Eq. (2), we denote

$$\sum_{m=1}^{M}\sum_{d=1}^{D}\sum_{s=1}^{S}\int_0^T E_q(\lambda(t, \mathbf{p}))dt = \sum_{k=1}^{K}\beta_k(b_k + \alpha b'_k) + \sum_{k'=1}^{K'}\omega_{k'}c_{k'},$$

where

$$\begin{aligned}
c_{k'} =& T\sum_{m=1}^{M}\sum_{n=1}^{N_m}\sum_{d=1}^{D}\sum_{s=1}^{S}\omega x'_k(q_{m,n,s,d}),\\
b_k =& \sum_{m=1}^{M}\sum_{n=1}^{N_m}\sum_{d=1}^{D}\sum_{s=1}^{S}\sum_{l=1}^{n-1}(x_{q_l(\mathbf{p}'),q_n}(t_n))_k((K(t_{m,n} - t_{m,l})\\
&- K(t_{m,n-1} - t_{m,l})),\\
b'_k =& \sum_{m=1}^{M}\sum_{n=1}^{N_m}\sum_{d=1}^{D}\sum_{s=1}^{S}\sum_{l=1}^{n-1}(x_{q_l(\mathbf{p}'),q_n}(t_n))_k\kappa(|\mathbf{p}' - \mathbf{p}_n|)).
\end{aligned}$$

Note that $K(t) = \int_0^t \kappa(s)ds$ and $\mathbf{p}' = (d, s)$. It is worth noting that to maximize the log-likelihood in Eqn (2), we need to iteratively update only the model parameters ω, β, and α. Therefore for computation efficiency, we precompute the likelihood according to the dimension of our utilized basic features and our proposed contextual features, and use the precomputed likelihood in the iterative procedure for model parameter optimization.

The optimization of ω and β are coupled due to the complex structure in $\log \lambda(t, \mathbf{p})$, which is usually computationally very expensive. To alleviate this limitation, we optimize a surrogate function (illustrated in Eqn 3), instead of the original $\log \lambda(t, \mathbf{p})$. Mathematically the employed surrogate function can be expressed as

$$\begin{aligned}
&g(\omega, \beta, \alpha|\omega^{(j)}, \beta^{(j)}, \alpha^{(j)}) = \eta_{m,n,k'} \log(\omega_{k'}x'_{k'}) \quad (3)\\
&+ \sum_{k=1}^{K}\eta_{m,n,K'+k}\log(\beta\sum_{l=1}^{n-1}\mathbf{x}_{q_l,q_n}(t_n)\kappa(t_{m,n} - t_{m,l}))\\
&+ \sum_{k=1}^{K}\eta_{m,n,K'+K+k}\log(\alpha\beta\sum_{l=1}^{n-1}\mathbf{x}_{q_l,q_n}(t_n)\kappa(|\mathbf{p}_{m.n} - \mathbf{p}_{m,l}|))\\
&- \sum_{k'=1}^{K'}\eta_{m,n,k'}\log(\eta_{m,n,k'}) - \sum_{k=1}^{K}\eta_{m,n,K'+k}\log(\eta_{m,n,K'+k})\\
&- \sum_{k=1}^{K}\eta_{m,n,K'+K+k}\log(\eta_{m,n,K'+K+k})\\
&- \sum_{k=1}^{K}\beta_k(b_k + \alpha b'_k) + \sum_{k'=1}^{K'}\omega_{k'}c_{k'}.
\end{aligned}$$

Note that $\{\eta\}$ is a set of branching variables formulated as

$$\begin{aligned}
\eta_{m,n,k'} &= \frac{\omega_{k'}x'_{k'}}{g},\\
\eta_{m,n,K'+k} &= \frac{\beta_k\sum_{l=1}^{n-1}\mathbf{x}_{q_l,q_n}(t_n)\kappa(t_{m,n} - t_{m,l})}{g},\\
\eta_{m,n,K'+K+k} &= \frac{\beta_k\sum_{l=1}^{n-1}\mathbf{x}_{q_l,q_n}(t_n)\alpha\kappa(|\mathbf{p}_{m.n} - \mathbf{p}_{m,l}|)}{g}.
\end{aligned}$$

where

$$g = \omega\mathbf{x}' + \beta\sum_{l=1}^{n-1}\mathbf{x}_{q_l,q_n}(t_n)(\kappa(t_{m,n} - t_{m,l}) + \alpha\kappa(|\mathbf{p}_{m.n} - \mathbf{p}_{m,l}|))$$

The surrogate function in Eqn 3 is derived based on the Jensen's inequality; it bounds $-\mathcal{L}(\mu, \beta, \alpha)$ from above and has a clear advantage of decoupling the optimization on ω and β. Moreover, by optimizing this surrogate function using the Majorize-Minimization (MM) algorithm [17], we can attain a global optimum (minimum) of $-\mathcal{L}$.

Interestingly we can interpret $\eta_{m,n,k'}$ as the probability that the n-th click event occurs due to the k'-th basic feature, $\eta_{m,n,K'+k}$ as the infectivity of all historical click events on the n-th click event of user m with regard to the k-th contextual feature and the temporal distance, while $\eta_{m,n,K'}$ is the infectivity of all historical click events on the n-th click event of user m with regard to the k'-th basic feature and the position distance.

3.1 Learning

We use a variational expectation-maximization (EM) algorithm [9] to compute the empirical Bayes estimates of the Hawkes hyperparameters ω, β, and α in our Hawkes model. As proved in [22], optimizing the surrogate function g as in Eqn (3) ensures that $\mathcal{L}$ decreases monotonically, thus guarantees that $\mathcal{L}$ will converge to a global optimum. Then by optimizing g, we are able to update ω, β, and α independently with closed-form solutions. Thus we propose to optimize the surrogate function as in Eqn (3) instead of the real likelihood, it iteratively approximates the posterior by fitting the variational distribution q, and optimizes the corresponding bound against the parameters ω, β and α independently with closed-form solutions as follows:

$$\beta_k = \frac{1}{b_k} \sum_{m=1}^{M} \sum_{n=1}^{N_m} (\eta_{m,n,K'+k} + \eta_{m,n,K'+K+k}),$$

$$\omega_{k'} = \frac{1}{c'_k} \sum_{n=1}^{N_m} \eta_{m,n,k'},$$

$$\alpha = \frac{1}{\sum_k b'_k} \sum_{m=1}^{M} \sum_{n=1}^{N_m} \eta_{m,n,K'+K+k}.$$

Complexity Analysis. The majority of our computation lies in the estimation of ω, β and α, where we need to calculate a vector of η for each event/QAC session n. Since feature-related computations such as $\beta \sum_{l=1}^{n-1} \mathbf{x}_{q_l,q_n}(t_n)\kappa(t_{m,n} - t_{m,l}))$, $c_{k'}$, b_k, b'_k, and $\alpha\beta_k \sum_{l=1}^{n-1} \mathbf{x}_{q_l,q_n}(t_n)\kappa(|\mathbf{p}_{m.n} - \mathbf{p}_{m,l}|)$ can be done ahead, the calculation of η in each iteration has a computational cost of $O(N * (2 * K + K'))$ only. Based on calculated η, the estimation procedure for ω, β and α also cost $O(N * (2 * K + K'))$. Notice that $N = \sum_{m=1}^{M} N_m$ is the total number of QAC sessions, K is the size of our proposed contextual features, and K' is the size of the basic features employed in this paper. Thus, our algorithm costs $O(N * (K + K'))$ in total per iteration, where $K \ll N$ and $K' \ll N$ can be ensured by controlling by the number of features we use. Since N is the total number of events/QAC sessions, we can view the computational cost as linear in the number of QAC sessions.

4. EXPERIMENTS

In this section, we evaluated our Hawkes model on real-world data sets, and compared its performance with the following representative baselines:

MPC [2, 31]: This MostPopularCompletion method, is a widely used baseline in Query Auto-Completion, and frequently adopted as one main feature in many QAC engines. This models utilized temporal information.

Table 2: Log Predictive Likelihood on Real-world Data

Data set	Platform/Model	Hawkes	RBCM	TDCM
OldQAC	PC	-167.21	-185.37	-208.65
OldQAC	Mobile	-150.58	-177.95	-195.04
NewQAC	PC	-137.64	-161.98	-179.72
NewQAC	Mobile	-122.97	-148.26	-162.40

UBM [11]: This User Browsing Model proposes a number of assumptions on user browsing behaviors, from which the probability of observing a document can be estimated. It depends on statistical counting of prefix-query pairs, thus unable to predict unseen prefix-query pairs. This model utilized the spacial information.

BSS [14]: This Bayesian Sequential State model uses a probabilistic graphical model to characterize the document content and dependencies among the sequential click events within a query with a set of descriptive features. This is a content-aware model, which can predict unobserved prefix-query pairs. This model utilized both spacial and temporal information.

TDCM [23]: This is a two-dimensional click model which emphasizes two kinds of user behaviors. It consists of two models, that is, a horizontal model which explains the skipping behavior, and a vertical model that depicts the vertical examination behavior. This model takes both the spacial and temporal information of user clicks into consideration.

RBCM [21]: This is a probabilistic model that solves the query auto-completion task by capturing three types of relationship between users' behaviors at different keystrokes in high-resolution QAC logs. This model considered both temporal and spacial information in user intent understanding.

4.1 Real-world Data

We conducted extensive experiments on two real-world high-resolution QAC logs that collected from the widely used Yahoo's search engine. The first data set, named *OldQAC*, contains high-resolution QAC logs from May 2014 to July 2014. The collection consists of a sample of 7.4 million QAC sessions from about 40,000 users over a 3-month period. The QAC sessions of each user span from 22 days to 3 months. According to the platform that each QAC session belongs to, we split the entire dataset into two subsets; one of them is collected from PC users, and it contains 4.6 million QAC sessions, while the other is collected from mobile device users, and it contains 2.8 million QAC sessions.

The second data set, named *NewQAC*, is also collected from Yahoo's search engine. It contains randomly sampled high-resolution QAC logs from Dec 2014 to Feb 2015. This QAC log contains 8.2 million QAC sessions, where 4.9 million sessions are from PC users, and the rest 3.3 million QAC sessions are from mobile users.

4.2 Experimental Results

We present the experimental results in the following subsections.

4.2.1 Model Fitness.

We evaluate the fitness of our proposed model on real-world data, and compare our model with probabilistic approach based methods, including RBCM and TDCM. All three competing models are designed to optimize the likelihood that search engine users click suggestions located at certain slots in each QAC session, which makes the comparison fair and meaningful. We split the data based on the time information: the QAC sessions occurred in the first 90% of the time period are used as the training data, while the remaining 10% are used as the test data. Table 2 shows the log

Table 3: Performance Comparison of QAC Methods. *Boldfaced results indicate p-value<0.05 compared to MPC. MB stands for the platform of Mobile.*

Data/Platform	Hawkes	TDCM	RBCM	MPC	UBM	BSS
Measured by MRR@Last						
OldQAC/PC	**0.694**	**0.592**	**0.608**	0.543	0.441	0.545
OldQAC/MB	**0.770**	**0.685**	**0.708**	0.649	0.431	0.650
NewQAC/PC	**0.732**	**0.602**	**0.642**	0.567	0.501	0.552
NewQAC/MB	**0.811**	**0.691**	**0.749**	0.631	0.482	0.654
Measured by MRR@All						
OldQAC/PC	**0.612**	**0.538**	**0.554**	0.464	0.467	**0.531**
OldQAC/MB	**0.671**	**0.611**	**0.629**	0.564	0.471	0.524
NewQAC/PC	**0.664**	**0.578**	**0.602**	0.522	0.508	**0.572**
NewQAC/MB	**0.754**	**0.628**	**0.676**	0.592	0.521	0.554

predictive likelihood on sessions falling in the final 10% of the total time of QAC log data. According to Table 2, Hawkes fits the real-world data much better than the competing probabilistic model based QAC methods, including both RBCM and TDCM. The results show that appropriate modeling of the influence across different QAC sessions is more important than capturing the relationship between users' behaviors within the same QAC session. RBCM fits the data better than TDCM; this observation shows that the relationship between users' behaviors is also useful for the prediction of search engine users' click choices among all suggestions under certain prefixes.

4.2.2 Query Auto-Completion.

To evaluate the effectiveness of the proposed model in suggesting users intended queries in each QAC session, we compare the proposed model with the state-of-the-art QAC algorithms. All compared methods re-rank the suggested queries at each keystroke and compete to rank the intended query as high as possible. Notice that among three probabilistic model based QAC methods, our proposed model and TDCM utilize the predicted probability that a search engine user clicks a suggestion located at a certain slot straightforwardly to rank the suggestions with a given prefix, while RBCM utilizes the relevance model (linear regression) part to rank the suggestion based on employed features, instead of using click probabilities directly. We employ the Mean Reciprocal Rank (MRR), a widely used measure for evaluating the QAC engine's performance [2, 31, 23], as the relevance metric. MRR can be denoted as

$$MRR = \frac{1}{|Q|} \sum_{q \in Q} \frac{1}{\mathrm{rank}_q},$$

where Q is the set of queries a user finally submitted, and rank_q denotes the rank of the query q in the suggested query list. Besides measuring the performance of QAC by MRR, we also perform significance tests using paired t-test with 0.05 as the p-value threshold.

Notice that among the suggested query lists of all keystrokes, those lists that do not contain users' finally submitted queries are removed from our experimental analysis. Since our experiments are conducted on high-resolution QAC data, we report both the average MRR score of all keystrokes, and the average MRR of the last keystroke only, since this is the keystroke where the user's click occurs. Notice that existing works which didn't make use of high-resolution QAC logs usually used the MRR of the last keystroke to measure their performance. In the following experiments, the whole dataset is divided evenly into a training set and a test set for different settings.

Table 3 compares the proposed model with state-of-the-art QAC algorithms by MRR. We can observe that Hawkes performs the best among all compared approaches, and outperforms existing QAC algorithms by over 11%, for all the data sets and different settings. Moreover, significance tests show that the differences between the proposed model and those baselines are statistically significant. The major difference between Hawkes and state-of-the-art QAC methods is that the proposed model is designed to capture the influence between users' click choices in each QAC session from three aspects: temporal, position, and contextual, while existing QAC methods generally ignored most or part of such influence. Thus, such phenomenon demonstrates the effectiveness of making use of the relationship between users' click behaviors across different QAC sessions in solving the QAC task, appropriate modeling of such relationships makes the proposed model significantly better than those alternative baselines which fail to utilize such relationships. Besides our proposed model, RBCM and TDCM perform better than the rest of existing QAC algorithms, which we attribute to the usage of high-resolution QAC logs. Here RBCM obtains a better result than TDCM, which may attribute to the modeling of the relationship between users' behaviors within the same QAC session. BBS outperforms UBM since it adopts the content-aware relevance model. MPC performs the worse, since it pays little attention to users' behaviors in QAC logs. By comparing with the performance using all the keystrokes and last keystroke only, we find that the advantages of the proposed model are ever more significant when measured by MRR@All. It indicates that the proposed model can recommend user intended queries higher with less keystrokes. The advantage of the proposed model on all the data sets and different settings also shows the robustness of the proposed model and the consistency of its improvement.

4.2.3 Strategy Selection.

In the following, we explore how different factors utilized in our model, including temporal, spacial, and contextual, contribute to the influence between click events in different QAC sessions. Since all three factors are critical in modeling the influence between users' clicks across different QAC sessions, and they closely collaborate in such influence modeling, we design this series of experiments to testify the absence of which factor can result in the greatest loss in the modeling of influence between clicks, instead of evaluate the effect of each factor in influence modeling alone. In the following experiments, we compare the proposed model using all three factors with those using only two factors among them. Those alternative solutions include:

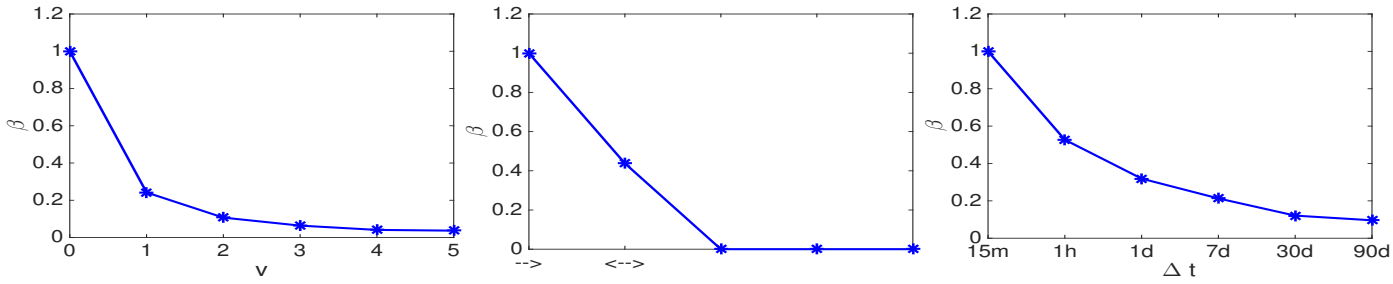
T & C: This model utilizes both temporal and contextual factors in modeling the influence between the click events in different QAC sessions. The spacial factors is not utilized in this model.

S & C: This model utilizes both spacial and contextual factors in modeling the influence between the click events in different QAC sessions. The temporal factors is not utilized in this model.

T & S: This model utilizes both temporal and spacial factors in modeling the influence between the click events in different QAC sessions. The contextual factors is not utilized in this model.

Also, we denote the Hawkes model that utilizes all three factors, i.e., temporal, spacial, and contextual, as **T & S & C**. Table 4 compares the performance of the proposed model with above alternative solutions in solving the query auto-completion task measured by MRR.

According to Table 4, **T & S & C** outperforms all three alternative solutions, which illustrates that all our utilized factors have a positive contribution to the modeling of influence between the clicks in different QAC sessions. Among all three alternative solutions, **T & S** performs the worst, which shows that contextual data play the most important role in capturing the influence between



(a) Query Distance (b) Temporal Order (c) History Range

Figure 2: Coefficient Learning of Contextual Features. The Y axis (β) denotes the average value of the learned coefficients of features characterized by different types. These average values are scaled to the range of [0, 1] to clarify the comparison of relative importance of different features. Here In Figure (a), features are categorized by the distance between two queries when issued. In Figure (b), features are categorized by whether the co-occurrence of two issued queries are counted with the temporal order taken into consideration or not. In Figure (c), features are categorized by the time range that the co-occurrence of queries are counted.

Table 4: Performance Comparison of Hawkes that Captures Influence of Different Factors

Data set	T & S & C	T & C	S & C	T & S
Measured by MRR@Last				
OldQAC/PC	0.694	0.658	0.632	0.611
OldQAC/MB	0.770	0.740	0.727	0.720
NewQAC/PC	0.732	0.711	0.691	0.652
NewQAC/MB	0.811	0.798	0.775	0.761
Measured by MRR@All				
OldQAC/PC	0.612	0.588	0.570	0.559
OldQAC/MB	0.671	0.649	0.638	0.634
NewQAC/PC	0.664	0.646	0.625	0.611
NewQAC/MB	0.754	0.719	0.698	0.682

click events. Such results can imply that, when provided with a set of suggestions, a user makes the click choice mainly based on his/her current search intent, which we believe is captured by our proposed contextual features. Meanwhile, **T & C** performs the best among all three alternative solutions, which illustrates that the spatial distance between users' clicks is the weakest signal in influence modeling. Such phenomenon shows that search engine users are not likely to take the slot where a suggestion is located as the top priority when they make a click decision. **S & C** performs worse than **T & C**, which emphasizes that instant situation also play a very important role in users' click choices. Although the temporal distance between click events alone does not influence users' click decision given a set of suggestions, it plays a very important role in understanding users' instant intent when works jointly with contextual data, which makes it more powerful in the proposed model than the position information.

4.2.4 Coefficient Learning of Contextual Features.

This series of experiments estimate the coefficients in our proposed model, and compare the difference of coefficients of different categories of features. Figure 2(a) shows the coefficients of features that characterized by v, the distance between two sequential queries. From Figure 2(a), we can find that with larger v, the coefficients of corresponding features shapely decrease. Such phenomenon implies that the relationship between two queries becomes significantly weaker with respect to the increase of distance that those two queries are issued in historical log. The coefficient of the feature that counts the co-occurrence of adjacent queries is significantly larger than the coefficients of rest features, which highlights the closeness of the relationship between queries that search engine users issue sequentially. Figure 2(b) shows the coefficients of features that characterized by the temporal order of the sequential queries that users issue. As introduced in Table 1 in above sections, "–>" indicates those features that take the temporal order of queries into consideration, while "<–>" indicates features that do not consider the temporal order information. From Figure 2(b), we can find that the coefficient of those features that count the co-occurrence of queries following the same temporal order is larger than that of features counting the co-occurrence of queries without taking the temporal order into consideration. We may conclude that search engine users do have some preference on the temporal order of queries they submit. Figure 2(c) shows the coefficients of features that characterized by the time range where the co-occurrence of queries occur. From Figure 2(c), we can find that when co-occurrence of queries in the latest 15 minutes is the most powerful signal for the prediction of the next query a user to issue given his/her current submitted query. Along with the enlargement of the time range, the signal becomes weaker. Such phenomenon may due to the fact that users' search intent changes from time to time. Given the same proceeding issued query, search engine users' click choices can vary with respect to different periods.

4.2.5 Case Study of Query Auto-Completion.

Now we show a few examples that illustrate how the proposed model recommends users queries that better satisfy user intent by capturing the influence between users' click behaviors across different QAC sessions. Figure 3 shows how the proposed model and RBCM, which performs the best among all state-of-the-art QAC methods in our experiments, rank the suggestions given various prefixes in a different way. The example presented in this figure is a QAC session where a user finally submitted "star wars", which is the user's intended query in this session. From Figure 3, we can find that the proposed model generally ranks users' intended queries higher than RBCM, especially at keystrokes with shorter prefixes. For example, with the prefix "st", Hawkes ranks the intended query at the position 1 while RBCM ranks it at the position 3. Although RBCM already utilizes the user's preference of the clicked queries at the last keystroke to improve their rankings at shorter keystrokes in the future by modeling the relationship between users' behaviors at different keystrokes, this method is still not sure about the user's specific intent in the current QAC session given a relatively short prefix, since queries such as "staples" and "stubhub" are also submitted frequently in the past. On the other hand, Hawkes jointly utilizes the temporal, spacial, and contextual factors in modeling the influence between users' suggestion click

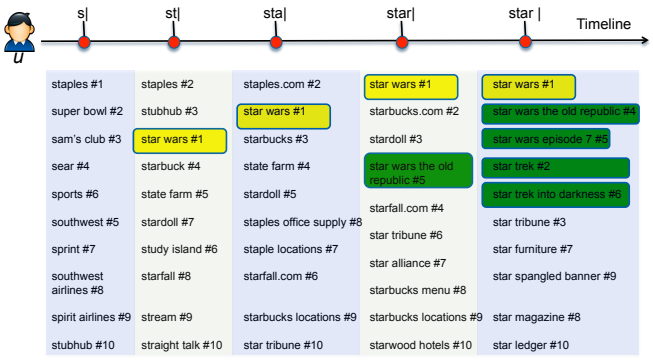


Figure 3: Case Study: The position of listed queries from top to down shows their rankings predicted by RBCM; the number tagged with # behind each query shows its ranking given by the proposed model. The yellow box highlights the user's intended query; the green box highlights queries satisfy similar user intent. Note that "|" denotes the cursor.

events, which can successfully understand this user's instant intent in the current QAC session. Since in the last QAC session, the user submitted the query "science fiction movie", based on the generated contextual features, we can guess that "star wars" probably satisfies his/her current intent given the entered prefix. Therefore, the proposed model ranks "star wars" at the top position. We also notice that queries of similar intent, such as "star wars the old republic" and "star wars episode 7" are ranked lower by our proposed model than by RBCM, which may due to that those queries are rarely issued in the history, which results in a low co-occurrence of those queries with the query "science fiction movie". Thus the proposed contextual features do not work under such situation. Meanwhile, we find that the suggestion "star trek" are ranked at a higher position by the proposed method than that by RBCM, since "star trek" are also closely related to "science fiction movie", and it is more popular than queries "star wars the old republic" and "star wars episode 7". Thus we can conclude that appropriate modeling of influence between users' click behaviors in different QAC sessions is critical for predicting users' instant intent given short prefixes under the current QAC session.

5. RELATED WORK

Hawkes Processes Our proposed model is closely related to point processes, which have been used to model social networks [4] and natural events [40]. People find self-exciting point processes naturally suitable to model continuous-time events where the occurrence of one event can affect the likelihood of subsequent events in the future. One important self-exciting process is Hawkes process, which is first used to analyze earthquakes [26, 40], and then widely applied to many different areas, such as market modeling [12, 1], crime modeling [33], terrorist [28], conflict [36], scholarly literature [35, 25], and viral videos on the Web [7]. To solve such models, an EM algorithm is generally adopted to estimate the maximum likelihood of Hawkes process [20].

Query Auto-Completion (QAC) The main objective of QAC is to predict users' intended queries and assist them formulate a query while typing. The most popular QAC algorithm is to suggest completions according to their past popularity. Generally, a popularity score is assigned to each query based on the frequency of the query in the query log from which the query database was built. This simple QAC algorithm is called MostPopularCompletion (MPC), which can be regarded as an approximate maximum likelihood estimator [2].

Several QAC methods [2, 32, 31, 34, 5] were proposed to extend MPC from various aspects. Bar-Yossef and Kraus [2] introduced the context-sensitive QAC method by treating users' recent queries as context and taking into account the similarity of QAC candidates with this context for ranking. But there is no consensus of how to optimally train the relevance model. Shokouhi [31] employed learning-based strategy to incorporate several global and personal features into the QAC model. However, these methods only exploit the final submitted query or simulate the prefixes of the clicked query, which do not investigate the users' interactions with the QAC engine.

In addition the above models, there are several studies addressing different aspects of QAC. For example, [32, 34] focused on the time-sensitive aspect of QAC. Other methods studied the space efficiency of index for QAC [3, 16]. Duan and Hsu [10] addressed the problem of suggesting query completions when the prefix is mis-spelled. Kharitonov et al. [19] proposed two new metrics for offline QAC evaluation, and [18] investigated user reformation behavior for QAC.

The QAC is a complex process where a user goes through a series of interactions with the QAC engine before clicking on a suggestion. As can be seen from the related work, little attention has been paid to understand the interactions with the QAC engine. Until recently, Li et al. [23] created a two-dimensional click model to combine users' behaviors with the existing learning-based QAC model. In this paper, we attempt to directly model and leverage the relationship between users' behaviors, so as to improve the performance of QAC.

Click Models This work is related to click models. In the field of document retrieval, the main purpose for modeling users' clicks is to infer the intrinsic relevance between the query and document by explaining the positional bias. The position bias assumption was first introduced by Granka et al. [13], stating that a document on higher rank tends to attract more clicks. Richardson et al. [29] attempted to model the true relevance of documents by imposing a multiplicative factor. Later examination hypothesis is formalized in [8], with a key assumption (Cascade Assumption) that a user will click on a document if and only if that document has been examined and it is relevant to the query. In addition, several extensions were proposed, such as the User Browsing Model (UBM) [11], the Bayesian Browsing Model [24], the General Click Model [39] and the Dynamic Bayesian Network model (DBN) [6]. Despite the abundance of click models, these existing click models cannot be directly applied to QAC without considerable modification. The click model most similar to our work is [38], which models users clicks on a series of queries in a session. However because of the main difference between QAC and document retrieval, our model is very different from [38].

6. CONCLUSION AND FUTURE WORK

We presented a novel Hawkes process based model for QAC by taking into account the context, temporal and position information. The model systematically captures the context influence between query pairs, along with the corresponding temporal and position influence. We evaluate our proposed methods on two real-world data in comparison with state-of-the-art methods. Experimental results show that our method offers a significant better performance comparing with state-of-the-art methods. The results also demonstrate that the proposed model can significant improve the performance by integrating different factors together. In the future work, it would be interesting to consider additional information such as location in the proposed model. Meanwhile, we plan to investigate alternative models that can be applied to other search tasks.

7. REFERENCES

[1] Y. Ait-Sahalia, J. Cacho-Diaz, and R. Laeven. Modeling financial contagion using mutually exciting jump processes. *Tech. rep.*, 2010.

[2] Z. Bar-Yossef and N. Kraus. Context-sensitive query auto-completion. In *WWW*, pages 107–116, 2011.

[3] H. Bast and I. Weber. Type less, find more: fast autocompletion search with a succinct index. In *SIGIR*, pages 364–371, 2006.

[4] C. Blundell, K. A. Heller, and J. M. Beck. Modelling reciprocating relationships with hawkes processes. *NIPS*, 2012.

[5] F. Cai and M. de Rijke. Selectively Personalizing Query Auto-Completion. In *SIGIR*, pages 993–996, 2016.

[6] O. Chapelle and Y. Zhang. A dynamic bayesian network click model for web search ranking. In *WWW*, pages 1–10, 2009.

[7] R. Crane and D. Sornette. Robust dynamic classes revealed by measuring the response function of a social system. *Proceedings of the National Academy of Sciences of the United States of America*, 105(41):15649–15653, 2008.

[8] N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *WSDM*, pages 87–94, 2008.

[9] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38, 1977.

[10] H. Duan and B.-J. P. Hsu. Online spelling correction for query completion. In *WWW*, pages 117–126. 2011.

[11] G. E. Dupret and B. Piwowarski. A user browsing model to predict search engine click data from past observations. In *SIGIR*, pages 331–338, 2008.

[12] E. Errais, K. Giesecke, and L. R. Goldberg. Affine point processes and portfolio credit risk. *SIAM Journal on Financial Mathematics*, 1(1):642–665, Sep 2010.

[13] L. A. Granka, T. Joachims, and G. Gay. Eye-tracking analysis of user behavior in www search. In *SIGIR*, pages 478–479. ACM, 2004.

[14] A. D. H. Wang, C. Zhai and Y. Chang. Content-aware click modeling. In *WWW*, pages 1365–1376, 2013.

[15] A. G. Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, 1971.

[16] B.-J. P. Hsu and G. Ottaviano. Space-efficient data structures for top-k completion. In *WWW*, pages 583–594, 2013.

[17] D. R. Hunter and K. Lange. A tutorial on mm algorithms. *The American Statistician*, Vol. 58, No. 1, pages 30–37, 2004.

[18] J.-Y. Jiang, Y.-Y. Ke, P.-Y. Chien, and P.-J. Cheng. Learning user reformulation behavior for query auto-completion. In *SIGIR*, pages 445–454, 2014.

[19] E. Kharitonov, C. Macdonald, P. Serdyukov, and I. Ounis. User model-based metrics for offline query suggestion evaluation. In *SIGIR*, pages 633–642, 2013.

[20] E. Lewisa and G. Mohlerb. A nonparametric em algorithm for multiscale hawkes processes. *Journal of Nonparametric Statistics*, 1, 2011.

[21] L. Li, H. Deng, A. Dong, Y. Chang, H. Zha, and R. Baeza-Yates. Analyzing user's sequential behavior in query auto-completion via markov processes. In *SIGIR*, pages 123–132, 2015.

[22] L. Li and H. Zha. Learning parametric models for social infectivity in multi-dimensional hawkes processes. In *AAAI*, 2014.

[23] Y. Li, A. Dong, H. Wang, H. Deng, Y. Chang, and C. Zhai. A two-dimensional click model for query auto-completion. In *SIGIR*, pages 455–464, 2014.

[24] C. Liu, F. Guo, and C. Faloutsos. Bayesian browsing model: Exact inference of document relevance from petabyte-scale data. *ACM Transactions on Knowledge Discovery from Data*, 4(4):19, 2010.

[25] X. Liu, J. Yan, S. Xiao, X. Wang, H. Zha, and S. Chu. On Predictive Patent Valuation: Forecasting Patent Citations and Their Types. In *AAAI*, 2017.

[26] Y. Ogata. Statistical models for earthquake occurrences and residual analysis for point processes. *Journal of the American Statistical Association.*, 83(401):9–27, 1988.

[27] P. O. Perry and P. J. Wolfe. Point process modeling for directed interaction networks. In *Journal of the Royal Statistical Society*, 2013.

[28] M. D. Porter and G. White. Self-exciting hurdle models for terrorist activity. *The Annals of Applied Statistics*, 6(1):106–124, 2011.

[29] M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: estimating the click-through rate for new ads. In *WWW*, pages 521–530, 2007.

[30] F. Schoenberg. Introduction to point processes. *Wiley Encyclopedia of Operations Research and Management Science*, pages 616–617, 2010.

[31] M. Shokouhi. Learning to personalize query auto-completion. In *SIGIR*, pages 103–112, 2013.

[32] M. Shokouhi and K. Radinsky. Time-sensitive query auto-completion. In *SIGIR*, pages 601–610, 2012.

[33] A. Stomakhin, M. B. Short, and A. L. Bertozzi. Reconstruction of missing data in social networks based on temporal patterns of interactions. *Inverse Problems.*, 27(11), Nov 2011.

[34] S. Whiting and J. M. Jose. Recent and robust query auto-completion. In *WWW*, pages 971–982, 2014.

[35] S. Xiao, J. Yan, C. Li, B. Jin, X. Wang, H. Zha, and X. Yang. On modeling and predicting individual paper citation count over time. In *IJCAI*, 2016.

[36] A. Z.-Mangion, M. Dewarc, V. Kadirkamanathand, and G. Sanguinetti. Point process modelling of the afghan war diary. *PNAS*, 109(31):12414–12419, July 2012.

[37] A. Zhang, A. Goyal, W. Kong, H. Deng, A. Dong, Y. Chang, C. A. Gunter, and J. Han. adaqac: Adaptive query auto-completion via implicit negative feedback. In *SIGIR*, pages 143–152, 2015.

[38] Y. Zhang, W. Chen, D. Wang, and Q. Yang. User-click modeling for understanding and predicting search-behavior. In *KDD*, pages 1388–1396, 2011.

[39] Z. A. Zhu, W. Chen, T. Minka, C. Zhu, and Z. Chen. A novel click model and its applications to online advertising. In *WSDM*, pages 321–330, 2010.

[40] J. Zhuang, Y. Ogata, and D. V. Jones. Stochastic declustering of space-time earthquake occurrences. *Journal of the American Statistical Association.*, 97(458):369–380, 2002.