Modeling the Influence of Popular Trending Events on User Search Behavior

Shubhra Kanti Karmaker Santu
Univeristy of Illinois at Urbana-Champaign
karmake2@illinois.edu

Liangda Li
Yahoo Research
liangda@yahoo-inc.com

Dae Hoon Park
Yahoo Research
daehoon@yahoo-inc.com

Yi Chang
Huawei Research America
yichang@acm.org

ChengXiang Zhai
Univeristy of Illinois at Urbana-Champaign
czhai@illinois.edu

ABSTRACT

Understanding how users' search behavior is influenced by real world events is important both for social science research and for designing better search engines for users. In this paper, we study how to model the influence of events on user queries by framing it as a novel data mining problem. Specifically, given a text description of an event, we mine the search log data to identify queries that are triggered by it and further characterize the temporal trend of influence created by the same event on user queries. We solve this data mining problem by proposing computational measures that quantify the influence of an event on a query to identify triggered queries and then, proposing a novel extension of Hawkes process to model the evolutionary trend of the influence of an event on search queries. Evaluation results using news articles and search log data show that the proposed approach is effective for identification of queries triggered by events reported in news articles and characterization of the influence trend over time, opening up many interesting opportunities of applications such as comparative analysis of influential events and prediction of event-triggered queries by users.

1. INTRODUCTION

While much work has been done on improving a search engine, little attention has been paid to how external factors influence the user search behavior, which is clearly important for improving a search engine. One important type of external factor is the trending events that "significantly" attract the general mass. Consider the following example. The hollywood movie "Captain America : Civil war" was released on May 6, 2016 and NYTimes published a review article [1] about the movie on the same day. To analyze how users search for this trending event, we collected two months (April and May, 2016) query log data from a well-known commercial search engine (https://search.yahoo.com/) and retrieved the top 500 unique queries relevant to the published NYTimes article using the BM25 [25], a state-of-the-art retrieval function. For these top 500 relevant queries, we plot their frequency distributions within the two months (April and May, 2016) in Figure 1. Here, the vertical red line indicates the release time of the movie (as well as the publication time of the NYTimes article about it). The x-axis represents "time in days" where the movie release time is set to be zero; the preceding and following days were set accordingly. The y-axis represents the corresponding frequency of the top 500 unique queries retrieved using BM25. From Figure 1, two things are evident. First, the user search activity suddenly increases near the time when the event occurred and second, the activity exponentially goes down as we move away from the origin. This confirms the fact that the "release" of the movie triggered a lot of user queries asking for relevant information, thus influenced user search behavior "significantly".

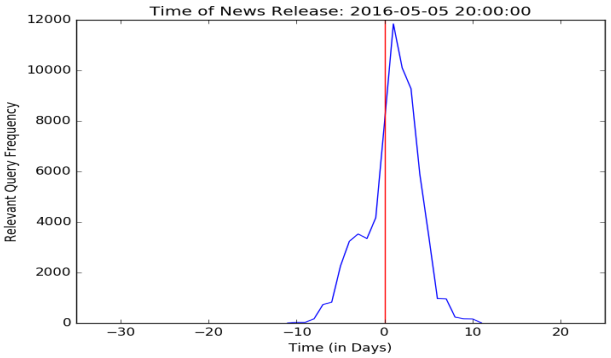

Figure 1: User search activity related to the release (May 6, 2016) of the Movie "Captain America Civil war".

How can we computationally model and analyze the influence of such trending events on user search behavior? What kind of queries are triggered by what kind of events? What kind of events tend to be most influential? How long does the influence last? Can we predict whether a user's query was triggered by a particular event? Besides being interesting social science research questions, these questions are also interesting from the perspective of improving the utility of a search engine. For example, if we can detect when a user's query is triggered by a particular event, it would help improve query auto-completion (by leveraging the terms occurring in news articles about the triggering event), improve

WWW'17 Companion, April 3–7, 2017, Perth, Australia.
ACM 978-1-4503-4914-7/17/04.
http://dx.doi.org/10.1145/3041021.3054188

the search results (by recommending relevant topics to the event), improve future query volume prediction, and detect influential events (which can then be recommended to users that might be searching for related topics).

To the best of our knowledge, the questions mentioned above have not been addressed in the existing work. In this paper, we conduct the first study of the problem of modeling the influence of trending events on users' search behavior from the perspective of data mining. Specifically, we frame the problem as a novel data mining problem where, given a text description of an event, we mine the search log data to identify queries that are triggered by it and further characterize the temporal trend of influence created by the same event on user queries. From data mining perspective, such a joint mining problem is novel and presents interesting challenges. First, "Influence" is an abstract concept and thus, it is not straight-forward how to measure influence. Second, how can we fully characterize the influence of an event on search queries? Finally, how should we formally define this data mining problem?

To tackle these challenges, we focus on studying how an event might have triggered queries from users. This restrictive perspective allows us to quantitatively measure "influence" based on the number of queries triggered by the event. To determine whether a query is triggered by an event, we propose to make the decision based on both textual similarity and temporal proximity between the clicked web documents by users filing the query and the text description of the event. To capture how the influence of an event on user queries evolve over time, we propose a formal model based on the Hawkes process to characterize the trend of influence.

We evaluate the proposed influence modeling methods using two sets of data-sets: 1) NYtimes articles: we collected the most read articles from NYtimes during two months span of April and May, 2016 using the NYtimes developers API to use them as trending events and 2) Query-log data: We also collected two months contemporary query log data from a widely used popular search engine (https://search.yahoo.com).

Evaluation results using these data sets show that the proposed approach is effective for identification of queries triggered by events reported in news articles and characterization of the influence trend over time. We further show that the proposed extended Hawkes model is useful in many ways, including improving the accuracy of predicting whether a newly entered query by a user was triggered by an event (which further enables a search engine to optimize its response to the query accordingly) and answering many interesting questions related to understanding the influence of events on queries.

In Summary, we make the following contributions in this paper: (1) We conduct the first study of modeling the influence of trending events on search queries and frame the problem as a new data mining problem. (2) We propose a computational method for measuring the influence of an event on a query and discovering triggered queries by an event. (3) We propose a novel extension of Hawkes process to model the influence trend of an event on user queries over time. (4) We propose a way to quantitatively evaluate an influence model using the task of predicting whether a newly entered query by a user has been triggered by some event, and show that both the method for measuring influence and the extended Hawkes process are useful for this prediction task, and they can be used immediately in a search engine to potentially customize the response of the search engine to a user's event-triggered query based on the event.

2. RELATED WORK

Search query logs have been extensively studied to understand user search behavior and provide better search experience [17, 21, 29]. Existing work mostly focused on the inference of users' search intent based on their own search habit and search history. On the other hand, our paper tries to model how user behavior on a search engine is influenced by external factors such as trending events.

Temporal Information Retrieval and Event Detection are two areas closely related to our work. While Event Detection has been studied vastly in the literature (see [3] for a recent survey), research interest on Temporal Information Retrieval has grown recently [7]. However, we want to emphasize that, neither of these is the intended goal of this study and our primary motivation is somewhat orthogonal. To be more specific, our work does not intend to study how time-sensitive information needs can be addressed [10, 4] or how users' information need change over time [20] or how to detect some events from social networks/news media [3]. Rather, given that some event has already been reported, we go one step further to investigate how the event may impact/influence the search behavior of the users.

The notion of event-based retrieval was introduced by Strötgen and Gertz [28] by returning events instead of documents. Zhang et al. [32] addressed the detection of recurrent event queries. Ghoreishi and Sun [13] introduced a binary classifier for detecting queries related to popular events. Kanhabua [19] extended the work [13] by enabling the classifier to detect less popular queries beside popular ones. However, all these approaches are supervised classification methods and largely depend on the quality of training labels provided by humans, whereas our approach is completely unsupervised.

Kairam et. al. [18] investigated the online information dynamics surrounding trending events, by performing joint analysis of large-scale search and social media activity. Matsubara et. al. [22] presented a new model for mining large scale co-evolving online activities. Pekhimenko et al. [24] designed a system named "PocketTrend" that automatically detects trending topics in real time, identified the search content associated to the topics, and then intelligently pushed this content to users' local machine in a timely manner. However, none of these studies provide answer to the question: how to model the temporal trend of influence created by an event on user queries, which is one of the primary motivations of our work.

Another important topic related to this paper is point process, which has been used to model social networks [5] and natural events [33]. 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 was first used to analyze earthquakes [33], and then widely applied to many different areas, such as market modeling [12], crime modeling [27], conflict [30], viral videos on the Web [9] etc. In this work, we extend the original Hawkes process to propose a new model that can capture the dynamics of influence by trending events on user search behavior.

3. PROBLEM FORMULATION

We solve the problem of modeling the influence of an event on search queries by framing it as a novel data mining problem where we would jointly mine two different types of data, i.e., text data describing many events and search log data that contains user queries and clickthrough records. We assume that each event, E, is represented as a tuple $<W_E,t_E>$, where, W_E is some natural text description of that event (e.g., some news article about that event) and t_E is the publication timestamp of W_E. A query is represented as a tuple with three attributes, i.e., $<W_q,t_q,U_q>$. Here, W_q is the set of keywords that query q contains, t_q is the timestamp of the query submission and U_q is the URL that the user clicked after posing the query. The desired output includes the following three elements:

1. Influential Events (or "Trending Events"): An influential event is any event E that attracts the interest of the general mass significantly and has triggered queries from many users. We want to discover a set of most influential events from the data sets.

2. Triggered Queries: A triggered query (denoted by q) by event E is a query entered by a user due to knowing information about event E. In other words, had the user not heard about E, he/she would not have entered query q. For each influential event E, we want to discover all the triggered queries by the event.

3. Influence Trend Model (or just Influence Model): An influence trend model for event E is a parameterized process that can model the temporal trend of influence by event E on user search queries. The parameter values of the model should be interpretable for characterizing how the intensity of the influence evolves over time.

The rationale of requiring the model to be parameterized with interpretable parameters is so that we can use the parameter values to obtain a concise quantitative summary of the dynamics of the influence from an event, which is essential for enabling many interesting applications of influence analysis (e.g., answering questions such as "how quickly does the influence intensity grow over time?" and "how quickly the influence disappears?")

Our problem formulation would enable many interesting applications, particularly for understanding what kind of events tend to have more significant influence on user queries, what kind of queries were triggered by a particular event, and how the influence evolves over time. Such analysis can be potentially configured to compare different kinds of events, similar events reported in different time periods or by different sources, and to compare different user groups to understand how different groups of users may have been influenced by the same event in different ways.

4. METHODS FOR INFLUENCE DISCOVERY AND CHARACTERIZATION

To solve the proposed influence mining problem, we need to complete three subtasks: 1) Discovery of events that have significantly influenced user queries, which we will refer to as influential events. 2) Discovery of queries influenced by any influential event, which we will refer to as event-triggered queries, or simply triggered queries. 3) Characterization of the temporal trend of the influence of an influential event on user queries over time. All these tasks are new tasks that are challenging due to the lack of labeled data for supervised learning. They have not been studied in the previous work, thus we do not have natural baseline methods to start with either. Below we present our proposed unsupervised method for solving all the three problems.

A careful analysis of these tasks suggests that subtask one and subtask two both rely on solving the basic problem of determining whether one event has influenced a query. Once we can do that, we would be able to quantify the influence of an event by counting how many queries are influenced by the event, and also easily obtain which queries are influenced by which event.

To solve subtask three, we propose to model the frequency of triggered queries over time with a temporal process model based on the Hawkes process, which assumes that the frequency of triggered queries at time t is a function of a base frequency λ_0 capturing the general popularity of this kind of queries, how quickly the influence decreases over time (captured by a parameter β), and the homogeneity of user search behavior over time (captured by a parameter α). The advantage of such a model is that by fitting the model to our observed frequency of triggered queries, we can obtain these meaningful parameters that are directly useful for characterizing the trend of influence.

4.1 Discovery of influential events and triggered queries

Our basic problem is the following: Given a query and an event, how can we know whether the event has influenced the query? Due to the complexity of the notion of influence, a completely rigorous definition of influence is nearly impossible. To make the problem more tractable, we propose three reasonable heuristics to guide us in designing a computational measure of influence. Specifically, given an event $E =< W_E, t_E >$ and a particular query submission $q =< W_q, t_q, U_q >$, we can reasonably make the following three assumptions about influence, which would help us design a function to computationally measure the influence of E on q. Here, we denote the content of the clicked-URL, i.e., $content(U_q)$ simply by W_U.

Assumption 4.1 (Query-Textual-Similarity)**.** *The higher the textual-similarity between W_E and W_q, the higher the chances that q is triggered/influenced by E. This assumption allows us to prune cases where the query is completely irrelevant to the event.*

Assumption 4.2 (Temporal-Similarity)**.** *The higher the temporal-similarity between t_q and t_E, the higher the chances that q is triggered/influenced by E. This assumption allows us to distinguish queries triggered by similar events in the past from those triggered by a current event.*

The Temporal-Similarity is important because Query-Textual Similarity alone is insufficient. For example, consider the two trending events "US election 2012" and "US election 2016". Now, if a user poses a query "US election", it is hard to tell which event actually triggered the query submission. However, if we know the timestamp of the query submission, we can better predict the triggering event. For example, if the query was posed in the year 2016, then with high probability, it was triggered by the event "US election 2016" as it was trending at that moment. On the other hand, if it was posed in the year 2012, probably the triggering event was "US election 2012". Thus, besides textual similarity, tempo-

ral similarity also plays an important role in predicting the influence.

Another useful piece of information that helps to verify whether some event E indeed influenced the submission of query q is the content of the URL which the user clicked after posing the query. If the content of the clicked-URL, i.e., U_q, is highly similar to the text description of event E, that means the user was actually looking for news about the same event. This, in turn, means that query q was influenced/triggered by event E. This gives us our third heuristic:

Assumption 4.3 (ClickedURL-Textual-Similarity)**.** *The higher the textual-similarity between W_E and W_U (W_U is the text of the clicked documents by users who entered query q), the higher the chances that E triggered/influenced q.*

Intuitively, we would like to design a measure that combines all the three heuristics so that a query-event pair would be scored high if (1) the query text is similar to the event text description, (2) the clicked documents are similar to the event text description, and (3) the time stamp of the query and that of the event are close. One way to combine them is to design a similarity/distance function for each of these three dimensions and combine the three functions into one single scoring function. Specifically, we use the the following scoring function to measure the influence:

$$F(E,q) = TxtSim(W_E, W_q) \cdot TmpSim(t_E, t_q) \cdot TxtSim(W_E, W_U) \quad (1)$$

We discuss these components in more detail below:

***TxtSim*(W_E, W_q)**: Similarity between query-document pair has been studied in the literature for a long time. One popular function from the literature is the "Okapi BM25" ranking function [25]. However, we could not use "Okapi BM25" directly for the major limitations discussed below.

First, each "event-text" (description of the event) usually contains a title and a body. The title often contains more important words pertaining to the event, while the body contains verbose details. It is thus necessary to put more emphasis on matching the title keywords first and then match the body details. The original "BM25" similarity function unfortunately does not provide such customizations. "BM25F" [31], an extension of BM25, handles this relative term weighting scenario, although "BM25F" is also not directly applicable to our problem setting for the following reason: To be able to compare the influence across different trending events, it is necessary that the "BM25F" score computed for different pairs of "event-text" and "query-text" be comparable. However, this is not the case because there is a large variance in the length of both "event-text" and "query-text". One might argue that, "BM25F" provides "Document Length Normalization" and "Relative Term Weighting", which should resolve the problem. But ones careful attention would reveal that "BM25F" is designed for a setting where the information need, i.e., the query is constant and only the document is varied to compute the similarity. But, in our case, both the document and query are variable and thus, we need both "document length normalization" and some kind of "query length normalization".

To address the two issues mentioned above, we use the following *modified* version of BM25 as the TxtSim function to fit our problem setting. Let, $W_E =< W_{E_1}, W_{E_2}, ..., W_{E_n} >$ be the "event-text" and $W_q =< W_{q_1}, W_{q_2}, ..., W_{q_n} >$ be the "query-text".

$$TxtSim(W_E, W_q) = \sum_{i=1}^{|W_E|} \frac{\omega(W_{E_i}).IDF(W_{E_i}).TF(W_{E_i}, W_q).(k_1+1)}{TF(W_{E_i}, W_q) + k_1.(1 - b + b.\frac{|W_q|}{avgql})}$$
$$\text{subject to} \sum_{i=1}^{|W_E|} \omega(W_{E_i}) = 1 \quad (2)$$

Note that, equation 2 is similar to the original "BM25" with the exception of the new term $\omega(W_{E_i})$ and the constraint that the weights must sum to 1. $\omega(W_{E_i})$ is essentially the weight of each n-gram in the "event-text" which reflects the importance of that particular n-gram with respect to the "event-text". $\omega(W_{E_i})$ allows the $TxtSim(W_E, W_q)$ to be comparable across different W_E and W_q as we enforce the constraint $\sum_{i=1}^{|W_E|} \omega(W_{E_i}) = 1$. Furthermore, one can easily set $\omega(W_{E_i})$ in such a way so that title n-grams get more weight than body n-grams as well as bigrams get more weight over unigrams and vice-versa. The specific weights we used for our experiments are mentioned in section 6. The IDF (inverse document frequency) and TF (term frequency) bear the usual meaning as in the original BM25 function.

***TxtSim*(W_E, W_U)**: $TxtSim(W_E, W_U)$ is basically the similarity between a pair of documents, in contrast with $TxtSim(W_E, W_q)$, which is the similarity between a query and document pair. $TxtSim(W_E, W_U)$ is almost similar to $TxtSim(W_E, W_q)$ with the exception that $TxtSim(W_E, W_q)$ contains only one ω (for W_E), while $TxtSim(W_E, W_U)$ contains two, i.e., ω_1 and ω_2, where ω_1 and ω_2 are weight distributions for the event-text (W_E) and clicked-url-content (W_U) respectively.

$$TxtSim(W_E, W_U) = \sum_{i=1}^{|W_E|} \omega_1(W_{E_i}) \cdot \omega_2(W_{E_i}) \cdot IDF(W_{E_i}) \cdot TF(W_{E_i}, W_U)$$
$$\text{subject to,}$$
$$\sum_{i=1}^{|W_E|} \omega_1(W_{E_i}) = 1 \ , \ \sum_{i=1}^{|W_U|} \omega_2(W_{U_i}) = 1 \ , \ \omega_2(W_{E_i}) = 0 \text{ if } E_i \notin W_U \quad (3)$$

The essence of equation 3 is that matching an n-gram which has high weights for both W_E and W_U contributes more to the similarity between W_E and W_U, whereas, n-grams having low weights for one/both of the articles contribute less to the similarity.

***TmpSim*(t_E, t_q)**: The $TmpSim$ function is expected to behave in the following way: if two events are far distant in time, their temporal similarity should be low; whereas, if they are close in time, the temporal similarity should be high. We also assume that the temporal similarity decreases exponentially as the distance in time increases. Below is an example of a function with such desired properties, where δ is the decaying parameter:

$$TmpSim(t_E, t_q) = e^{-\delta.|t_E - t_q|} \quad (4)$$

4.2 Influence Trend Modeling

Once we can measure the influence an event E has over different user queries, the next task is to model the trend of such influence over time. Such modeling would allow us to study the characteristics of influence in a systematic way and enable many interesting applications like predicting future volume of queries, optimizing search recommendations etc. To accomplish this, we propose function $Trend(E, t)$,

which takes as input an event E and timestamp t and returns the popularity/trendiness of event E at timestamp t. There are many ways one can define how to measure the popularity/trendiness of an event. For example, the number of tweets related to the event, number of views for news articles relating to the event, click counts for the event webpage, number of social media posts sharing the event etc. In this work, we define popularity/trendiness of an event by the users tendency to pose queries that are relevant to the event. We choose this definition because we are specifically interested in modeling the influence of trending events on user search behavior.

Defining the $Trend(E, t)$ is not trivial. First, we introduce a set of assumptions that will help us formalize the notion of "trendiness".

Assumption 4.4 (Influence Growth)**.** *Each query submitted to search engine τ that is relevant to event E increases the chance of subsequent submission(s) of relevant queries to τ, thus, grows the trendiness/influence of event E.*

Assumption 4.4 simply says that each relevant query submission from one user indicates an increase in the tendency of other users to pose similar relevant queries. In other words, the trendiness of an event is directly/indirectly influenced by the previous query submissions relevant to the same event, which in turn, reflects the tendency to receive new queries relevant to the event. To see the rationale of Assumption 4.4, consider a scenario when a user is exposed to an event that he/she feels interested about, the user may use multiple queries to find out more about the event and then share the event details on some social media platform or talk to some friends about the event. These friends, being interested in the event after hearing about it, may do further search. Thus, the influence of the event propagates from one user to another and reflects in their search activity. As another example scenario, say some popular news portal publishes a featured article about some event. Many people would then read that article to know about the event/incident and start searching for more details about it. In this case, new incoming queries searching about a particular event gives useful indication about further submission of similar relevant queries. Thus, a significant number of relevant queries actually indicate the growth of the trendiness/influence of the event.

Given that the "trendiness" of an event, E, at moment t is dependent on all the relevant (w.r.t. E) queries posed before timestamp t, the next obvious question is: to what extent each of the previous queries contribute to the current "trendiness"? To answer this, we make two further assumptions as mentioned below.

Assumption 4.5 (Query Relevance)**.** *The contribution of a query q (submitted at time t_q) to the "trendiness" of an event E at moment t, where $t > t_q$, is proportional to the textual similarity between the "event-text", W_E and the "query-text", W_q, i.e., $TxtSim(W_E, W_q)$.*

Assumption 4.6 (Query Timestamp)**.** *The contribution of a query q (submitted at time t_q) to the "trendiness" of an event E at moment t, where $t > t_q$, exponentially decays as the difference between t and t_q increases.*

Assumption 4.5 and 4.6 are very reasonable. Assumption 4.5 basically says that, highly relevant queries grow the trendiness of an event, thus, indicates the growth in the volume of future relevant queries; at the same time, Assumption 4.6 says that the contribution of a past query to the current "trendiness" of the event decays exponentially with time.

4.2.1 A parametric model for influence

Incorporating these two assumptions, we introduce Equation 5 presented below to compute the "trendiness" of an event E at time t.

$$Trend(E, t) = \lambda_0 + \sum_{i=1}^{n} \alpha \cdot TxtSim(W_E, W_i) \cdot e^{-\beta(t-t_i)} \quad (5)$$

Equation 5 contains three parameters, i.e., λ_0, α and β. λ_0 is a constant which reflects the base trendiness that is assumed to be always present. α is a scaling factor to control the contribution of $TxtSim(W_E, W_i)$ on the current "trendiness" and β is the scaling factor to control the exponential decay in time. W_1, W_2, ..., W_n represents all the queries relevant to event E that were posed before timestamp t.

Equation 5 is not entirely new; it is similar to the self-exciting point processes [11] e.g. Hawkes Process [16]. However, the point processes models the recurring events of the same type, whereas our task is to model influences of one type of event (e.g. Trending articles) on other type of events (e.g. user search behavior). Thus, we include the textual-similarity between the past queries and event-text, i.e., $TxtSim(W_E, W_i)$, into the basic Hawkes process to fit our problem scenario.

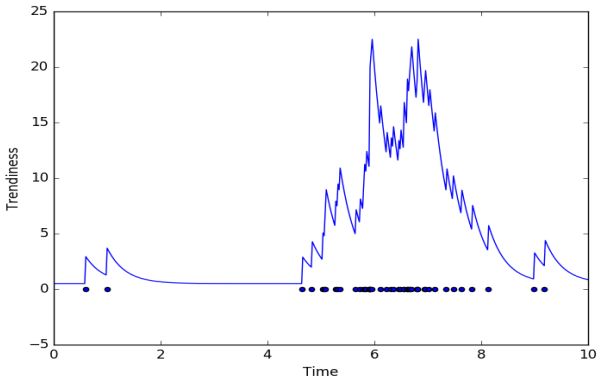

Figure 2: Simulating the trend of some hypothetical event.

Figure 2 shows a hypothetical simulation of how equation 5 works. Without loss of generality, we assume that $TxtSim(W_E, W_i) = 1.0$ for any choice of W_E and W_i. However, this choice does not affect our attempt to present the spirit of equation 5. The x-axis in Figure 2 represents time and the y-axis represents the corresponding "trendiness" of some hypothetical event E. The "Blue" dots represent each query submission that is relevant to E. These "Blue" dots were generated by simulating the Hawkes Process (see [23] for details). For this particular simulation, λ_0, α and β were set to 0.5, 2.5 and 3.0 respectively. This means, there is always a base trendiness of λ_0 equal to 0.5. The "trendiness" goes up as people start querying about the event E (note the first blue dot), which, increases the chance of generating further queries. Thus, the volume of queries influenced by E and "trendiness" of E grows mutually by enhancing each other. For example, at the vicinity of time zone $t = 6$, a lot of queries were posed which resulted in further query submissions and the "trendiness" rises up significantly. The "trendiness" goes down exponentially with time if no further

queries are posed. This is signified by the exponential decay near time zone $t = 8$. Note that, the current estimate of "trendiness" is directly correlated to the expected volume of relevant queries in near future. The higher the current "trendiness" is, the more the probability of observing high volume of relevant queries in future. However, the estimate of "trendiness" is incrementally updated as we move forward in time and observe (do not observe) new user queries.

4.2.2 Estimation of the parameters

Let the set of parameters be $\Lambda = \{\lambda_0, \alpha, \beta\}$. We adopt maximum likelihood estimation technique to find the optimal parameter values for equation 5. First, we show how to compute the log-likelihood for a single event E and then we extend it to multiple events case. Considering the query submission sequence q_1,q_2, ..., q_n (all related to event E) as a simple point process, the likelihood for a single trending event can be written as follows (see [23] for background):

$$\log L = \int_0^{t_{q_n}} (1 - Trend(E,t))dt + \sum_{i=1}^{n} \log(Trend(E, t_{q_i})) \tag{6}$$

After some simple mathematical operations, equation 6 boils down to the following form:

$$\log L = -\left(\frac{\alpha}{\beta}\right) \cdot TxtSim(W_E, W_{q_i}) \cdot \left\{1 - e^{-\beta \cdot (t_{q_n} - t_{q_i})}\right\} + t_{q_n} - \lambda_0 \cdot t_{q_n} + \sum_{i=1}^{n} \log(Trend(E, t_{q_i})) \tag{7}$$

Given the close form of the log likelihood function (equation 7), the optimization problem to find the optimal parameter set Λ^* is written as follows:

$$\Lambda^* = \arg\max_{\Lambda} L(\{q_1, .., q_n\}|E, \Lambda) \tag{8}$$

For multiple events $E_1, E_2, ...E_m$, the optimization problem is extended in the following way:

$$\Lambda^* = \arg\max_{\Lambda} \sum_{j=1}^{m} L(\{q_{j1}, .., q_{jn}\}|E_j, \Lambda) \tag{9}$$

One can use any non-linear optimization method to solve this maximization problem. Nelder-Mead Simplex Method [14] is one such popular optimization technique. Another useful approach is the Sequential Least SQuares Programming (SLSQP) [6].

5. EXPERIMENTAL DESIGN

Data Sets: We collected two sets of data sets: one for trending events and one for user query history. We call these two data sets *Trending-Event* dataset and *Query-Log* dataset respectively. The following two paragraphs provide details about these two data-sets:

Trending-Event data-set: An obvious choice for a text data set describing events is news articles (though other data such as social media might also be applicable). The NYTimes Developers Network (thanks to them) provides a very useful api called *"The Most Popular API"* [2], which automatically provides the *url's* of the most e-mailed, most shared and most viewed articles from NYTimes.com during the last month from the date of the issue of the query. We chose to use this API because of two major benefits: 1) it automatically removes duplicate articles, thus we don't need to deal with cases where multiple articles are related to the same event. 2) it only provides the most popular articles from NYTimes, thus the quality/accuracy of the events represented by these articles is very high. Using this API, we collected the most e-mailed, most shared and most viewed articles from the two months span: April and May, 2016. Each article consists of a tuple <*title-text*, *body-text*, *timestamp*>. Among different categories of news, we used only four categories for our experiments: *US (National Affairs)*, *Movies*, *Sports* and *World (International Affairs)*. Table 1 shows some details about the data-set.

Category	# of articles	Avg. Title Length	Avg. Body Length
Movies	25	18.88	458.08
Sports	15	19.53	508.4
US	18	20.38	487.77
World	11	18.18	438.81
Total	69	19.30	473.69

Table 1: Description of Trending-Event data set

Category	Total query	% Pos. instance	% Neg. instance	Avg. txt-sim
Movies	193,282	16.24	83.75	2.49
Sports	616,449	0.84	99.15	2.48
US	204,926	33.72	66.27	1.99
World	22,197	7.68	92.3	1.96
Total	1,036,854	10.35	89.64	2.38

Table 2: Description of Query-Log data set

Query-Log data-set: To analyze the user queries contemporary to the articles in *Trending-Event* data-set, we use the two-months (April and May, 2016) user query log data from the widely used search engine at https://search.yahoo.com. Each query submission q is represented as a tuple <*query-text*, *timestamp*, *clickedURL*>. The two-months query log data contains $105,925,732$ query submissions in total. To keep the computation feasible, for each article E in the *Trending-Event* data-set, we retrieved top 500 unique (in terms of text) queries that has at least a similarity score of 1.5 (with respect to E) according the textual similarity function in equation 2 and discarded the rest. This filtering step is reasonable because if the textual similarity is very low (less than 1.5), we assume that the influence prediction problem becomes trivial, i.e., there is no influence of E on the query. Thus, textual similarity itself is sufficient in this case to decide whether there is an influence or not. However, more challenging cases are when the query shares a high degree of textual similarity to the event E, but still is not influenced by the event. In this paper, we focus on these type of queries with significant textual similarity to the event and assume that the other queries are not influenced by any event in our data set. The summary of this data-set is presented in Table 2.

Predictive modeling for quantitative evaluation: Quantitative evaluation of the mining results pose challenges because there is a lack of gold standard for what events are influential and the ground truth for the true influence trend of an event. We overcome this difficulty by proposing a way to perform indirect quantitative evaluation based on the task of predicting whether a user's newly input query is triggered

by an event. The prediction setup is intended to simulate a real application scenario when a search engine receives a query from a user. In such a scenario, it would be beneficial for the search engine to "know" whether this query was triggered by a particular event since if it was, then the search engine would be able to leverage this knowledge to optimize the search results to be presented to the user (e.g., recommending content related to this event).

With such a setup, we can use the component techniques in our proposed mining approach, including text similarity functions, temporal similarity function, and the extended Hawkes model, to construct a prediction model to attempt to predict whether a "new" query in a separate held-out search log data set is influenced by any event based solely on the query without using any clickthrough information. The clickthrough information, however, is only used to create the gold standard labels for the evaluation purposes, i.e., whether such a query is indeed triggered by an event (we used equation 1 from section 4.1 along with threshold 0.01). One can argue that a better way to create the gold standard labels is to involve human judgments. However, for our data-set, this means the human annotators would have to go through $1,318,359$ <event, query, url-content> triplets, which is practically infeasible. So, we had to opt for some automated techniques for annotating the gold standard labels.

To evaluate the quality of the gold standard labels created by our automatic approach, we randomly sampled 200 positive and 200 negative examples labeled by the automatic process. Then, we asked three volunteers to independently go through these 400 <event, query, url-content> triplets and manually label each of them with 1 if, after reading the event description and contents of "url-content", the annotator thinks the query was indeed influenced by the event or 0, otherwise. We computed Cohen's kappa coefficient [8] to measure the inter-rater-agreement which was found to be reasonably high, i.e., 0.835. Thus, we conclude that the gold labels created by our automatic approach is reliable.

The labeled data-set created in the way previously described is highly imbalanced as most of the queries are not influenced with respect to some particular event E. To make the data-sets balanced, we randomly under-sampled from the pool of the negative samples to match the size of the positive examples for reporting the results in section 6. Concretely, we can use the following equation (eqn. 10) to compute the influence relation between event E and query q without using the clickthrough information and then pick a reasonable threshold to separate the influenced queries from the rest. We did threshold analysis which showed that the prediction model remains very stable for wide range of threshold value, i.e., $[0.8, 5]$ and we chose 1.0 for our experiments (the details are omitted due to lack of space). We call this model the "IP" (Influence Prediction) model.

$$F(E,q) = TxtSim(W_E, W_q) \cdot TmpSim(t_E, t_q) \cdot Trend(E, t_q)) \quad (10)$$

Performance Metric: To evaluate the performance of the proposed predictive model, i.e., IP, we use the four popular measures available in the literature: *precision*, *recall*, *specificity* and *F-measure* (see [15] for details). We also present results for the recently introduced K-measure [26] to show that "IP" model achieves better performance in terms of this new measure too.

	Bigram	**Unigram**	**Sum**
Title	0.49	0.21	0.70
Body	0.21	0.09	0.30
Sum	0.70	0.30	1.00

Table 3: Weight allocation for Title vs Body and Unigram vs Bigram in equation 2.

6. RESULTS

In this section, we report our experimental findings including both qualitative and quantitative evaluation results. We first start with some implementation details, then describe the qualitative and quantitative evaluation sequentially.

6.1 Implementation Detail

For the weight distribution ω in equation 2, we followed the weighting scheme presented in Table 3. This significance of Table 3 is that it puts more weight on the title-text matching (0.7) in comparison to the body-text matching. Similarly, it puts more weight on bigram-matching (0.7) in comparison to unigram-matching (0.3). An immediate consequence is that, bigram-matching in the title-text gets the highest reward (0.49), whereas, unigram-matching in the body-text gets the least reward (0.09). In other words, the weights for all bi-grams in the title of the event-text summed up to 0.49 and the weight of each individual bi-gram is proportional to its term frequency in the title-text. Similarly, the weights for all unigrams in the body of the event-text summed up to 0.09 and the weight of each individual unigram is proportional to its term frequency in the body-text. The same weighting scheme was used for $\omega_1(W_{E_i})$ and $\omega_2(W_{E_i})$ in equation 3.

The "trendiness" parameters, i.e, $\Lambda = \{\lambda_0, \alpha, \beta\}$ are learnt automatically using equation 9 (see table 6 for the exact values). Parameter δ (equation 4) was heuristically set to 0.8. This heuristic value is not an issue because for all the variants of the "IP" model (mentioned in table 7), we use the same δ value, thus, the optimal value is irrelevant for comparative analysis.

6.2 Qualitative evaluation

We show some sample data mining results to analyze their quality. First, we show the top four most influential events for each category in our data set as measured by the overall/total number of triggered queries in Table 4. They are all intuitively influential events. For example, the top one for category "Movies" is the release of "Caption America : Civil War", while the top one for category "World" is the "Panama Papers leaked".

Second, we show a sample of triggered queries with highest frequency for the event "curt schilling fired from espn" and event "panama papers leaked" in Table 5. We see that these queries are indeed well associated with these events.

Finally, we examine the optimal parameters learned by fitting the modified Hawkes model in Table 6.

Interpreting Model Parameters: We focus on interpreting the optimal values of the modeling parameters, $\Lambda = \{\lambda_0, \alpha, \beta\}$, which are automatically learnt by the estimation technique introduced in section 4.2.2. Table 6 shows these values. Indeed, these values have intuitive interpretation that matches our real-life expectation. For example, λ_0 essentially reflects the general interest in posing queries related

#	Movies	Sports	US	World
1	"captain America: Civil War" released	san antonio spur vs oklahoma city thunder basketball	harriet tubman ousts andrew jackson	panama papers leaked
2	alden ehrenreich Cast "hail caesar"	Rise of leicester city in premiere league	donald trump comments on transgenders toilet use	sadiq khan elected in london
3	gosling and Crow star "nice guys"	curt schilling fired from espn	donald trumps running mate	philippine presidential election
4	ken loach wins Palme dor	western conference finals	indiana primary elections	brazil president impeachment

Table 4: Top influential events for different categories.

#	curt schilling fired from espn	panama papers leaked
1	curt schilling espn	panama paper leak
2	espn curt schilling suspension	celebrity involved in panama offshore account
3	curt schilling facebook post	panama paper politicians
4	curt schilling comment	panama paper american
5	curt schilling blog	panama paper law firm

Table 5: Popular queries triggered by influential events.

Category	λ_0	α	β
Movies	0.0420	0.9082	2.6539
Sports	0.0146	0.4810	1.0208
US	0.0656	1.0892	2.3727
World	0.0117	0.1464	0.3292

Table 6: Trendiness parameter for different types of events.

to some trending event. Table 6 shows that people usually have the most interest ($\lambda_0 = 0.0656$) in the US category, i.e., events related to the national affairs. While the international affairs, i.e., "World" category generally draws the least attention ($\lambda_0 = 0.0117$). Next, α models the degree of homogeneity in user search behavior. So, high value of α means high degree of similarity in the intra-community search pattern. For example, in case of the category "US" ($\alpha = 1.0892$) and "Movies" ($\alpha = 0.9082$), we found the homogeneity to be significantly higher than for the category "Sports" ($\alpha = 0.4810$) or "World" ($\alpha = 0.1464$), indicating the search behavior is more diverse and discrete for the "Sports" and "World" category. Finally, β models the decay in user interest with time; thus, high value of β indicates a quick drop of interest among the general mass. As expected, β obtained for the "Movies" category ($\beta = 2.6539$) was found to be the highest, as people usually talks a lot about movies when they get released and the topic disappears quickly in few days. However, to our surprise, we also obtained high value of β for the "US" category ($\beta = 2.3727$). One plausible explanation for this fact may be that there are too many national news to follow and people switch their interest from time to time following different national news. On the other hand, value of β for the "Sports" ($\beta = 1.0208$) and "World" ($\beta = 0.3292$) category was found to be smaller, indicating the general interest is somewhat more prevailing in these cases. All these results show some qualitative analysis about how effective our proposed mining model is.

Capturing Trend: To verify how well our proposed extended Hawkes model can capture the trend of influence by some event on user queries, we generated the simulated "trendiness" plot (Figure 3) for the event "Captain America : Civil War" (The real data is plotted in Figure 1). To generate Figure 3, we used the learnt optimal parameters from table 6, i.e., $\{\lambda_0, \alpha, \beta\} = \{0.0420, 0.9082, 2.6539\}$. We also assumed that at a certain moment t, we know all the past queries that were influenced by the event along with their textual similarity to the event-text. Then, we used equation 5 to compute $Trend(E, t)$ at different t and plotted that in Figure 3. Cross examination of Figure 3 and 1 reveals that, the extended Hawkes model can, in fact, capture the real trend quite reasonably.

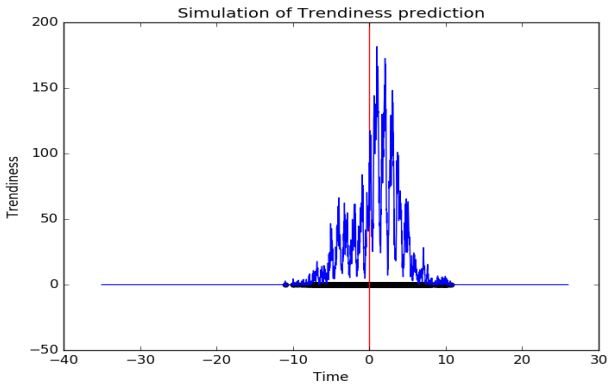

Figure 3: Simulation of Trendiness for the event: release of "Captain America : Civil War" Movie

Our qualitative analysis thus shows that overall the proposed approach is able to generate meaningful and interesting knowledge that can help better understand the influence of news events on user queries.

Method	TxtSim	TmpSim	Trend
txt	Yes	No	No
txt-time	Yes	Yes	No
txt-trnd	Yes	No	Yes
txt-time-trnd	Yes	Yes	Yes

Table 7: Summary of different versions of the "IP" model.

6.3 Quantitative evaluation with predictive modeling

We now turn to quantitative evaluation of the proposed approach using the predictive modeling task for predicting whether a user's newly input query has been influenced by an event. To better understand the role of each three basic components of the "IP" model (see equation 10), we create four different versions of our model by throwing out one or more components at a time and using the rest of the components to predict the influence of events on user queries. Table 7 shows these different versions of IP model along with the components it contains. For example, the "txt" method only contains the "TxtSim" component, while "txt-trend" contains both "TxtSim" and "Trend" components, but does not incorporate the "TmpSim" component. From now onwards, we refer to all the methods compared in this paper by the terminology introduced in table 7 to report the experimental results. All results reported in this section

Category	Method	F-measure	K-measure	Precision	Recall-f	Recall-K	Specificity
Movies	txt	0.6667	0.2366	0.5282	0.9035	0.5684	0.6682
	txt-time	0.7089	0.2717	0.5505	0.9952	0.8656	0.4061
	txt-trnd	0.8084	0.5526	0.6978	0.9606	0.9068	0.6457
	txt-time-trnd	**0.8087**	**0.5532**	0.6987	0.9598	0.9045	0.6488
Sports	txt	0.6667	0.1482	0.5000	1.0000	0.7626	0.3855
	txt-time	0.6783	0.2584	0.5132	1.0000	0.5634	0.6950
	txt-trnd	**0.8239**	0.5990	0.7345	0.9382	0.9382	0.6609
	txt-time-trnd	**0.8239**	**0.5992**	0.7348	0.9378	0.9378	0.6615
US	txt	0.6710	0.0372	0.5063	0.9945	0.0952	0.9420
	txt-time	0.7441	0.5234	0.6444	0.8804	0.6806	0.8428
	txt-trnd	0.8120	0.5758	0.7291	0.9161	0.9161	0.6597
	txt-time-trnd	**0.8155**	**0.5878**	0.7380	0.9112	0.9112	0.6766
World	txt	0.6680	0.0205	0.5018	0.9988	0.0393	0.9812
	txt-time	0.6759	0.4560	0.5111	0.9977	0.5252	0.9308
	txt-trnd	**0.8198**	**0.6020**	0.7489	0.9056	0.9056	0.6964
	txt-time-trnd	**0.8193**	0.6014	0.7493	0.9039	0.9039	0.6975

Table 8: Prediction Results for different IP models

used equation 2 as the "TxtSim" component. We also experimented with other text-similarity functions, e.g., TF-IDF cosine similarity; however, the results turned out to be significantly poor (more than 10% relative difference in F-measure) as compared to using equation 2 [the details are omitted due to lack of space].

Table 8 shows the summary of the performance obtained by the four different versions of "IP" model on the *Trending event* and *Query-Log* Dataset. For each method and event-category, the table reports the F-measure (with corresponding Precision and Recall, i.e., Recall-f) and and K-measure (with corresponding Specificity and Recall, i.e., Recall-K). Each result reported in Table 8 is the average of 25 runs using five-iterated-five-fold cross validation, each time with a random initialization of the parameter set $\{\lambda_0, \alpha, \beta\}$. It is evident that for all the categories of events, the "txt-time-trnd" method performs the best in terms of both F-measure and K-measure. For example, in case of the category "Movies", the "txt-time-trnd" method obtains a F-measure and K-measure value of 0.8087 and 0.5532 respectively, while the only textual similarity based method, i.e., "txt" achieves a F-measure and K-measure value of 0.6667 and 0.2366 respectively.

Close observation of Table 8 reveals that, only textual similarity is not sufficient for the influence prediction task as demonstrated by the relative poor performance of the "txt" method. Adding the "TmpSim" component, i.e., "txt-time" improves the prediction accuracy, although not to a significant degree. However, adding the "Trend" component to the "TxtSim" component results in a significant jump in the prediction accuracy ("txt-trnd" method) which verifies that the "Trend" component is very important to detect the influence of events on user query submissions. For example, for the "US" category, "txt" obtains a F-measure value of 0.6710 and "txt-time" obtains 0.7441, whereas, "txt-trnd" obtains a F-measure value of 0.8120. Finally, combining all the three components, i.e., "txt-time-trnd" achieves slightly better performance (F-measure 0.8155) than "txt-trnd" (F-measure 0.8120) signifying the fact that, once we have incorporated the "Trend" component, there is little room for "TmpSim" to further improve the prediction performance. This verifies our assumption that, "Trend" is an essential component for this kind of influence prediction task.

Overall, these quantitative evaluation results show that the basic component techniques we proposed for modeling influence, i.e., text similarity, temporal similarity, and extended Hawkes model, are all useful for the prediction task, suggesting that they indeed capture useful signals for modeling the influence relation of events on user queries. It is especially interesting to note that the modified Hawkes model provides a "trendiness" score that is shown to be beneficial for the prediction task, suggesting that the model has indeed captured the trend of influence well.

7. CONCLUSION AND FUTURE WORK

In this paper, we conducted the first study of how trending events influence search queries, where we frame the problem as a novel data mining problem of joint mining of trending event news data and search log data. We proposed a computational method to quantitatively measure the influence of an event on a query and to discover queries triggered by the event. Specifically, we proposed a novel extension of Hawkes process to model the evolutionary trend of the influence of an event on search queries. Evaluation results show that our proposed approach effectively identifies queries triggered by events and characterizes the influence trend of different types of events.

Although we mainly applied the proposed model to the problem where we predict if a query was triggered by an event, our model can be applied to many other problems. For example, it can help query auto-completion by leveraging terms related to the triggering event, and it can also improve search results by boosting documents that are relevant to the event. In addition, analysis on different characteristics of the events can enable us accurately detect more influential events. All of these interesting directions are left as future work.

8. ACKNOWLEDGMENTS

This work was done as part of an intern research project at Yahoo Research. This work is also supported in part by a Yahoo Faculty Research and Engagement Program award. We thank the five anonymous reviewers for their valuable feedback to help us improve the paper.

9. REFERENCES

[1] Review: In 'captain america: Civil war', super-bro against super-bro. http://www.nytimes.com/2016/05/06/movies/captain-america-civil-war-review-chris-evans.html?_r=0. Accessed: 2016-07-30.

[2] The most popular api: Nytimes developers network. https://developer.nytimes.com/most_popular_api_v2.json\#/README, 2016. Accessed: 2016-07-30.

[3] F. Atefeh and W. Khreich. A survey of techniques for event detection in twitter. *Computational Intelligence*, 31(1):132–164, 2015.

[4] K. Berberich, S. Bedathur, O. Alonso, and G. Weikum. A language modeling approach for temporal information needs. In *European Conference on Information Retrieval*, pages 13–25. Springer, 2010.

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

[6] P. T. Boggs and J. W. Tolle. Sequential quadratic programming. *Acta numerica*, 4:1–51, 1995.

[7] R. Campos, G. Dias, A. M. Jorge, and A. Jatowt. Survey of temporal information retrieval and related applications. *ACM Computing Surveys (CSUR)*, 47(2):15, 2015.

[8] J. Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46, 1960.

[9] 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.

[10] W. Dakka, L. Gravano, and P. Ipeirotis. Answering general time-sensitive queries. *IEEE Transactions on Knowledge and Data Engineering*, 24(2):220–235, 2012.

[11] D. J. Daley and D. Vere-Jones. *An introduction to the theory of point processes: volume II: general theory and structure*. Springer Science & Business Media, 2007.

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

[13] S. N. Ghoreishi and A. Sun. Predicting event-relatedness of popular queries. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 1193–1196. ACM, 2013.

[14] R. Glaudell, R. T. Garcia, and J. B. Garcia. Nelder-mead simplex method. *Computer Journal*, 7:308–313, 1965.

[15] C. Goutte and E. Gaussier. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In *ECIR*, pages 345–359. Springer, 2005.

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

[17] J. Jiang, D. He, and J. Allan. Searching, browsing, and clicking in a search session: changes in user behavior by task and over time. In *ACM SIGIR*, 2014.

[18] S. Kairam, M. Morris, J. Teevan, D. Liebling, and S. Dumais. Towards supporting search over trending events with social media. In *International AAAI Conference on Web and Social Media*, 2013.

[19] N. Kanhabua, T. Ngoc Nguyen, and W. Nejdl. Learning to detect event-related queries for web search. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1339–1344. ACM, 2015.

[20] A. Kulkarni, J. Teevan, K. M. Svore, and S. T. Dumais. Understanding temporal query dynamics. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 167–176. ACM, 2011.

[21] L. Li, H. Deng, A. Dong, Y. Chang, and H. Zha. Identifying and labeling search tasks via query-based hawkes processes. In *ACM SIGKDD*, 2014.

[22] Y. Matsubara, Y. Sakurai, and C. Faloutsos. The web as a jungle: Non-linear dynamical systems for co-evolving online activities. In *Proceedings of the 24th International Conference on World Wide Web*, pages 721–731. ACM, 2015.

[23] T. Ozaki. Maximum likelihood estimation of hawkes' self-exciting point processes. *Annals of the Institute of Statistical Mathematics*, 31(1):145–155, 1979.

[24] G. Pekhimenko, D. Lymberopoulos, O. Riva, K. Strauss, and D. Burger. Pockettrend: Timely identification and delivery of trending search content to mobile users. In *WWW*, 2015.

[25] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, M. Gatford, et al. Okapi at trec-3. *NIST SPECIAL PUBLICATION SP*, 109:109, 1995.

[26] F. Sebastiani. An axiomatically derived measure for the evaluation of classification algorithms. In *SIGIR ICTIR, 2015*.

[27] 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.

[28] J. Strötgen and M. Gertz. Event-centric search and exploration in document collections. In *Proceedings of the 12th ACM/IEEE-CS joint conference on Digital Libraries*, pages 223–232. ACM, 2012.

[29] R. W. White, W. Chu, A. Hassan, X. He, Y. Song, and H. Wang. Enhancing personalized search by mining and modeling task behavior. In *WWW*, 2013.

[30] 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.

[31] H. Zaragoza, N. Craswell, M. J. Taylor, S. Saria, and S. E. Robertson. Microsoft cambridge at trec 13: Web and hard tracks. In *TREC*, volume 4, pages 1–1, 2004.

[32] R. Zhang, Y. Konda, A. Dong, P. Kolari, Y. Chang, and Z. Zheng. Learning recurrent event queries for web search. In *Proceedings of the EMNLP 2010*, pages 1129–1139. Association for Computational Linguistics, 2010.

[33] 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.