Hindawi
Mathematical Problems in Engineering
Volume 2017, Article ID 7490879, 8 pages
https://doi.org/10.1155/2017/7490879

Research Article

Exploiting Query's Temporal Patterns for Query Autocompletion

Danyang Jiang, Honghui Chen, and Fei Cai

Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Hunan, China

Correspondence should be addressed to Danyang Jiang; danyangjiang@nudt.edu.cn

Received 18 September 2016; Accepted 5 March 2017; Published 23 March 2017

Academic Editor: Emilio Insfran

Query autocompletion (QAC) is a common interactive feature of web search engines. It aims at assisting users to formulate queries and avoiding spelling mistakes by presenting them with a list of query completions as soon as they start typing in the search box. Existing QAC models mostly rank the query completions by their past popularity collected in the query logs. For some queries, their popularity exhibits relatively stable or periodic behavior while others may experience a sudden rise in their query popularity. Current time-sensitive QAC models focus on either periodicity or recency and are unable to respond swiftly to such sudden rise, resulting in a less optimal QAC performance. In this paper, we propose a hybrid QAC model that considers two temporal patterns of query's popularity, that is, periodicity and burst trend. In detail, we first employ the Discrete Fourier Transform (DFT) to identify the periodicity of a query's popularity, by which we forecast its future popularity. Then the burst trend of query's popularity is detected and incorporated into the hybrid model with its cyclic behavior. Extensive experiments on a large, real-world query log dataset infer that modeling the temporal patterns of query popularity in the form of its periodicity and its burst trend can significantly improve the effectiveness of ranking query completions.

1. Introduction

Query autocompletion (QAC) provides a list of queries to search engine users from the moment that they start entering a query. This popular feature of search engines is intended to reduce the physical and cognitive effort when formulating a query [1]. As illustrated in Figure 1, upon a user's keystroke, QAC provides a ranked list of query completions that strictly start with the typed prefix. The user can submit a completed query by clicking any of them or continue to type the entire query. Typically, the query prefix tends to be short and ambiguous, making it difficult to capture user's search intent and to recommend relevant queries. Hence, the primary objective of a QAC system is to satisfy the average user typing the same prefix [1], where the most common and effective approach is introduced to exploit the query logs and rank the query completions according to their past popularity [2]. Although such approach can generate satisfactory query completion lists on average, it is far from optimal since it fails to take into account temporal patterns, which can potentially provide valuable signals to generate better QAC rankings. Although the query's recency has been taken into account in a QAC ranking model [1], the performance is still dissatisfying because this model assumes that query's future popularity distribution will remain the same as that previously observed. In contrast, in this paper, two temporal patterns of query's popularity will be exploited by considering both periodicity and burst trend.

As we know, for some queries, their popularity keeps relatively stable over time (e.g., *Google*, *Amazon*, and *Wikipedia*) or exhibits a periodic pattern (e.g., *Christmas*, *Mother's Day*, and *movie*); other queries may experience a sudden rise in popularity that is unexpected by previous observations, for example, *Belgium terrorist attack*. Typically this popularity surge is precipitated by some ongoing real-life events or breaking news that attract sudden peaks in public attention. We define such a sharp rise in query popularity as *burst*. Therefore, the QAC system should respond quickly to the unexpected spikes in query popularity and rank query completions accordingly. Previous works [3, 4] mainly focus on detecting the cyclic pattern of query popularity and rerank completions by their forecasted popularity, disregarding the

Figure 1: Illustration of QAC in a commercial search engine for the query "music download."

burst pattern of aperiodic queries. Our work tends to remedy the aforementioned problems.

In this paper, we propose a time-sensitive hybrid QAC ranking model that combines the periodicity and the burst trend of query popularity. We first employ the Discrete Fourier Transform (DFT) to detect the periodicity of a query's past popularity to forecast its future popularity by which we can produce an initial ranking of query completions. We then extend it by examining the burst trend of each query's popularity to generate a final ranking of query completions. The empirical experiments on a real-world query log dataset show that our proposal can improve the ranking performance over the state-of-the-art time-sensitive QAC baseline in terms of Mean Reciprocal Rank (MRR) by 3.5%.

We enumerate our main contributions as follows:

(1) We tackle the challenge of QAC in a novel way by considering both the periodicity and the burst trend of query popularity.

(2) We present a novel method to predict a query's future popularity which results in a better ranking of query completions.

(3) We analyze the effectiveness of our hybrid QAC model and find that it can significantly outperform the state-of-the-art time-sensitive QAC method in terms of MRR.

2. Related Work

2.1. Query Autocompletion. Generally, a QAC process consists of two steps: filtering and ranking [4, 5]. The first step is to generate matched query completions to user's input prefix and store them in an efficient indexing data structure such as tries for fast lookup. The second step is to sort these completions based on various ranking signals. To obtain the query completions in the filtering step, most QAC methods rely on the query logs [1, 2, 4, 5]. As a complement, Bhatia et al. [6] propose an unsupervised probabilistic approach to generate query completions from the document corpus instead of the query logs by selecting terms and phrases that are highly correlated to user's input prefix. In addition, in order to overcome the problem of mistyping, Chaudhuri and Kaushik [7] capture the typing errors by examining the edit distance and design an error-tolerant QAC strategy. Due to an enormous volume of query completions, space-efficiency is also important in the filtering process. Hsu and Ottaviano [8] present three indexing data structures with different space/time/complexity tradeoffs to deal with such efficiency related problems.

In contrast, in the ranking step, the most simple and straightforward ranking approach is based on query's past popularity, which is referred to as the Most Popular Completion (MPC) model [2]:

$$\mathrm{MPC}(p) = \arg\max_{q \in C(p)} w(q), \quad w(q) = \frac{f(q)}{\sum_{i \in Q} f(i)}, \tag{1}$$

where $C(p)$ represents a set of query completions that match the typed prefix p and $f(q)$ is the past popularity of query q in the search log Q.

Despite its straight-forwardness, MPC ignores the temporal patterns of query popularity and generates a uniform list of query completions in every situation across various searchers. Therefore, its performance is less than optimal. Recent works have been focused on adding more detailed information on query popularity into the ranking process to extend the MPC model. Shokouhi and Radinsky [4] develop a time-sensitive approach which orders query completions by the forecasted popularity using the time-series analysis of query history. In addition, Whiting and Jose [1] meliorate (1) by computing $w(q)$ using the query popularity evidence within time windows of recent 2–28 days.

Although the time-sensitive QAC approaches have been studied in [1, 3, 4], they do not consider the burst trend of short-term search history. Our proposal differs from these previous works in that we try to combine the periodicity with the burst trend to predict query's future popularity, thereby making full use of the temporal patterns of each query's popularity and generate effective QAC rankings.

2.2. Temporal Information Retrieval. As the content of the web changes constantly over time, search engines are required to offer the latest information to their users. Thus, exploiting temporal information to improve the freshness as well as the relevance of returned results has attracted extensive attentions in recent years in the field of information retrieval. Li and Croft [9] and Diaz and Jones [10] pioneer the temporal language models by incorporating the document creation dates into the language models. Due to the dynamic property in web search, identifying changes in query popularity and document content is an essential part in retrieval process. Kulkarni et al. [11] classify query popularity changes and examine the relationship between changes in query popularity, search result content, and query intent. Elsas and Dumais [12] propose two complementary methods to leverage document content change characteristics in ranking algorithms and show that considering document dynamics can yield performance improvements on the relevance ranking of navigational queries.

Understanding user's temporal search intent and disambiguating time expressed in queries are also challenging. Berberich et al. [13] inject the time information hidden in queries into a query-likelihood model to rank documents. Their work is extended by Kanhabua and Nørvåg [14] which determines the time of implicit temporal queries using top-ranked documents. In addition, Campos et al. [15] develop a

language-independent model that uses the temporal similarity measure to date implicit temporal query with time periods from returned web snippets.

After aggregating the query occurrence within a fixed time interval, the query volume can be seen as a time sequence. Therefore, time-series analysis methods could be applied to model the temporal changes in query popularity and forecast the future trend. Shokouhi [16] uses time-series decomposition techniques to identify seasonal queries. Vlachos et al. [17] build a time-series for each query and phrase in the query logs and then detect significant periodicities and bursts in sequences. Strizhevskaya et al. [3] measure the accuracy of various time-series prediction algorithms on the Yandex query log.

So far, the freshness and the recency have been taken into account in web search tasks. Extensive works have been done for mining query's underlying temporal patterns. To the best of our knowledge, this is the first piece of work where the periodicity and the burst trend of query popularity are combined for the task of QAC.

3. Approach

In this section we first formulate the problem of query autocompletion (QAC) and then describe our approach to rank query completions based not only on the forecasted popularity by periodicity but also on the short-term burst trend.

We formally define the QAC problem as follows: given a search log Q and a set of query completions $C(p)$ that matches the typed prefix p, a QAC system is able to rank query completions in $C(p)$ using ranking signals available at time t. The query completions set is dynamically updated with each new character entered by the user. The user can choose any query completion in $C(p)$ or type the entire query if none of the completions satisfies his need.

3.1. Periodicity-Based QAC Model. To detect the periodicity of a query's popularity, we first aggregate the query popularity within every day. By doing so, we can obtain a time-series of occurrence counts at specific days for each query. To detect the periodicity of a sequence, Fourier transform and autocorrelation are commonly applied [18]. However, autocorrelation is not a good indicator for the true period because the true period and its multipliers have high autocorrelation values, which results in difficulties to use a cutoff to determine the true period. In contrast, Fourier transform is able to accurately detect short to medium length periods. In addition, it has the merits of denoising and compression [19]. Furthermore, our experimental dataset collects 3-month query logs, in which queries at most possess short to medium periodicities. Thus, we choose the Discrete Fourier Transform (DFT) to find the periodic behavior of each query's popularity.

Fourier transform maps a signal of time into a new signal whose argument is frequency. In the case of a periodic sequence, the Fourier transform can be simplified to the calculation of a discrete set of complex amplitudes, called Fourier series coefficients. Let a time-series $f(q)_n$, where $n = 1, 2, \ldots, N$, denote the history popularity of query q in past N days; the normalized DFT of $f(q)_n$ is a series of complex numbers $F(q)_k$:

$$F(q)_k = \frac{1}{\sqrt{N}} \sum_{n=1}^{N} f(q)_n e^{-j(2\pi/N)kn}, \quad k = 1, 2, \ldots, N. \tag{2}$$

In order to find the underlying periodicity of $f(q)_n$ we use the periodogram to estimate the power spectral density of this sequence. The periodogram $P(k)$ is calculated as the squared magnitude of each Fourier coefficient:

$$P(k) = \left\| F(q)_k \right\|^2, \quad k = 1, 2, \ldots, \left\lfloor \frac{N}{2} \right\rfloor, \tag{3}$$

where $\|\cdot\|$ is the L2-norm. Due to the circular even symmetry property of the sequence $P(k)$, we can detect the frequencies that are at most half of the maximum signal frequency [20]. By assuming $P(k^*)$ is the maximum over all periodogram values of other frequencies, the frequency k^* has the strongest power in signal. Mapping frequency back to the time domain, frequency k^* corresponds to a period of $\lceil 1/k^* \rceil$ days which is the most dominant periodicity of query q's popularity.

We then use the detected periodicity of query q's past popularity to forecast its future popularity $\widehat{f}(q)_{t+1}$ at time $t+1$ by averaging its recent M observations at preceding time points $t + 1 - 1 \cdot T, \ldots, t + 1 - M \cdot T$ in the log:

$$\widehat{f}(q)_{t+1} = \frac{1}{M} \sum_{m=1}^{M} f(q)_{t+1-m \cdot T}, \tag{4}$$

where $T = \lceil 1/k^* \rceil$ denotes the periodicity of q's popularity. The QAC ranking model that uses $\widehat{f}(q)_{t+1}$ to calculate $w(q)$ in (1) is referred to as *P-QAC* hereinafter.

3.2. Burst Detection in Query Popularity. As the P-QAC model ranks query completions by their predicted popularity based on the periodicity, it ignores the burst trend of query's popularity. To mitigate this problem, we propose detecting the short-term burst of query popularity and predict its future trend at time $t + 1$.

For discovering the burst trend in a time-series of query popularity, Moving Average (MA) is commonly adopted [17, 21]. A burst emerges at time $t+1$ when its MA value $\mathrm{MA}(q)_{t+1}$ exceeds the cutoff point *cutoff* and the amplitude of this burst $\mathrm{amp}(q)_{t+1}$ can be formulated accordingly:

$$\mathrm{amp}(q)_{t+1} = \mathrm{MA}(q)_{t+1} - cutoff. \tag{5}$$

To limit reasonable fluctuation, *cutoff* is usually set to

$$cutoff = \mu_A + \gamma \cdot \sigma_A, \tag{6}$$

where μ_A and σ_A are the mean and standard deviation of MA sequence, respectively, and γ is a factor that smoothes out noise and highlights evident peaks.

As to the calculation of $\mathrm{MA}(q)_{t+1}$ in a time sequence, previous works [17, 21] employ Simple Moving Average (SMA) which treats the contribution of each previous point

equally and computes the unweighted mean in a certain sliding window for the whole sequence:

$$\mathrm{SMA}(q)_i = \sum_{n=i-L}^{i-1} \frac{1}{L} \cdot f(q)_n, \quad i = L+1, \ldots, t+1, \tag{7}$$

where L represents the length of the sliding window and $f(q)_n, n = 1, 2, \ldots, t$, is the time-series of query q's popularity prior to time t. However, in the field of temporal information retrieval, it is widely accepted that the recent values of a sequence matter more than the distant ones, and this difference can be captured by introducing a decay function on past values [14, 22]. Similarly, we introduce decay into our method and identify burst based on the computation of Weighted Moving Average (WMA):

$$\mathrm{WMA}(q)_i = \sum_{n=i-L}^{i-1} \mathrm{norm}(\omega_n) \cdot f(q)_n, \quad i = L+1, \ldots, t+1. \tag{8}$$

The normalized weight $\mathrm{norm}(\omega_n)$ controls the weight of each observation in the sliding window and ensures $\sum_n \mathrm{norm}(\omega_n) = 1$. A decay function is introduced before normalizing: $\omega_n = DecayRate^{i-n}$, where $i - n$ indicates the time interval between the future day i and the previous day n. The length of sliding window may affect the overall performance of QAC, which is to be discussed in our experiments later.

3.3. Hybrid QAC Model. To consider a query's long-term and short-term search history when predicting its future popularity, we propose a hybrid QAC ranking model that combines both the periodic and the burst patterns of query's popularity. First, we use P-QAC to produce a ranked list of completions $C(p)$ to the input prefix p. Then, we assign scores $P\mathrm{score}(q)$ and $B\mathrm{score}(q)$ to each completion based on the forecasted popularity and the burst amplitude, respectively. Finally, each completion is assigned to a final raking score $H\mathrm{score}(q)$ which is a convex combination of $P\mathrm{score}(q)$ and $B\mathrm{score}(q)$:

$$H\mathrm{score}(q) = (1-\lambda) \cdot P\mathrm{score}(q) + \lambda \cdot B\mathrm{score}(q), \tag{9}$$

where $0 \le \lambda \le 1$ is a tradeoff controlling the weights of the periodic score and the burst score.

In addition, the query completions in $C(p)$ may present different burst amplitudes. For instance, some queries show significant bursts in popularity and thus a uniform λ value can not capture such evidences. Therefore, a flexible value λ^* which varies according to the burst amplitude of each completion is introduced to replace λ in (9):

$$\lambda^* = \begin{cases} 0 & \mathrm{amp}(q)_{t+1} < \mu_B, \\ \lambda & \mathrm{amp}(q)_{t+1} \ge \mu_B, \end{cases} \tag{10}$$

where μ_B is the mean of burst amplitudes of query completions in $C(p)$. Considering that $P\mathrm{score}(q)$ and $B\mathrm{score}(q)$ use different units and scales, they need to be standardized before being combined. $P\mathrm{score}(q)$ is standardized as

$$P\mathrm{score}(q) = \frac{\widehat{f}(q)_{t+1} - \mu_P}{\sigma_P}, \tag{11}$$

where μ_P and σ_P are the mean and standard deviation of forecasted popularity of completions in $C(p)$. $B\mathrm{score}(q)$ is standardized in a similar manner:

$$B\mathrm{score}(q) = \frac{\mathrm{amp}(q)_{t+1} - \mu_B}{\sigma_B}, \tag{12}$$

where σ_B is the standard deviation of burst amplitude $\mathrm{amp}(q)_{t+1}$ of completions in $C(p)$. The details of our hybrid QAC model are described in Algorithm 1. The hybrid QAC model employs SMA or WMA to detect burst which is referred to as *H-SMA* or *H-WMA*. In addition, a hybrid QAC model with a fixed value λ used in (9) is denoted as λ*-H-WMA* or λ*-H-SMA* while a flexible value λ^* corresponds to λ^**-H-WMA*.

4. Experimental Setup

This section gives the details of our experimental setup. First, we list four research questions in Section 4.1 to guide our experiments. Then, we describe the dataset and the experimental parameters in Section 4.2. Finally, we briefly introduce our evaluation metric and the baselines in Section 4.3.

4.1. Research Questions. We address the following research questions in the remainder of the paper:

(RQ1) How does our periodicity-based QAC model P-QAC compare against the state-of-the-art time-sensitive QAC baseline? (See Section 5.1.)

(RQ2) Does the burst trend in our hybrid QAC model help improve the performance over P-QAC? (See Section 5.2.)

(RQ3) How are the impacts of scenarios for burst detection, that is, Weighted Moving Average (WMA) and Simple Moving Average (SMA), on QAC performance? (See Section 5.3.)

(RQ4) How does the length of sliding window affect the ranking effectiveness of our hybrid QAC model? (See Section 5.4.)

4.2. Dataset and Parameters Settings

4.2.1. Dataset. We use the publicly available AOL query log dataset [23] in our experiments. This dataset comprises sampled queries submitted by anonymized users to the AOL search engine from March 1, 2006, to May 31, 2006, which is sufficiently large to ensure statistical significance. We remove the navigational queries containing URL substring (.com, .net, .org, .edu, .mil, .gov, www., and http) and discard queries starting with special characters such as #, $, &, and @. After cleaning, the dataset consists of 7,402,014 distinct

Input: Search log Q; Query's past popularity $f(q)_n$; Number of returned completions K; Length of sliding window L;
Output: Ranking list of top K completions of p;
(1) **for** each $q \in Q$ **do**
(2) $T \leftarrow \mathrm{DFT}(f(q)_n)$;
(3) Calculate $\widehat{f}(q)_{t+1}$ by (4);
(4) **end for**
(5) **for** each $q \in Q$ **do**
(6) **for** each prefix p of q **do**
(7) Return top K completions of p (i.e., $C(p)$) ranked by $\widehat{f}(q)_{t+1}$;
(8) **end for**
(9) **end for**
(10) **for** each $q \in C(q)$ **do**
(11) Calculate $P\mathrm{score}(q)$ based on (11);
(12) Calculate $\mathrm{amp}(q)_{t+1}$ by (5);
(13) Calculate $B\mathrm{score}(q)$ based on (12);
(14) Calculate $H\mathrm{score}(q)$ based on (9);
(15) **end for**
(16) Rerank $C(p)$ by $H\mathrm{score}(q)$;
(17) **Return** a re-ranked list of $C(p)$;

Algorithm 1: Hybrid QAC model.

queries submitted by 559,721 unique anonymous users. It is then split into a training set and a test set with a ratio of 75%/25%. Traditional k-fold cross-validation is not applicable to streaming settings since it would disorder the temporal data sequence [24]. Therefore, queries in the training set were submitted before May 8, 2006.

Additionally, only queries appearing in both two sets are kept. In total, there are 456,010 unique queries in the training and test sets. Among them, 98,825 queries (22%) show periodic behavior and near 208,362 queries (46%) present burst behavior on average.

4.2.2. Parameters Settings. To smooth the popularity predicted by periodicity, that is, to determine the value of M in (4), we average all pervious observations of query's popularity at each periodic day in the search log due to the limited length of our dataset. Following [14], we use an exponential decay function with $DecayRate = 0.5$ in (8). Typical values for γ in (6) are set to [1.5, 2]; we set $\gamma = 1.8$, as it performs best in our trials. For obtaining an optimal tradeoff balance in our hybrid QAC model, we set $\lambda = 0.5$ in (9), which has been proved to be the best choice in a hybrid model [2]. In our experiments, an initial ranking of top 20 query completions is returned and to be reranked (i.e., $K = 20$).

4.3. Evaluation Metric and Baselines

4.3.1. Evaluation Metric. As a ranking task, we employ the Mean Reciprocal Rank (MRR) to evaluate the QAC performance. MRR serves as a standard evaluation metric in measuring the ranking accuracy of QAC task [1, 2, 5]. It is the average reciprocal rank in the list of query completions:

$$\mathrm{MRR} = \frac{1}{|S|} \sum_{q \in S} \frac{1}{\mathrm{rank}_q}, \tag{13}$$

where S denotes the testing set and rank_q is the rank of the final submitted query q in the query completion list. If no matched query is found in the query completion list, $1/\mathrm{rank}_q$ is set to 0.

4.3.2. Baselines. To verify the effectiveness of our proposal, the following competitive methods are adopted as baselines: (1) the Most Popular Completion (MPC) model that ranks query completions by their aggregated occurrences in the whole training period [2]; we refer to this model as *MPC-ALL* hereafter; (2) the state-of-the-art time-sensitive QAC approach that ranks query completions according to their past popularity within a fixed time window [1], which is denoted as *MPC-W*.

5. Results and Discussion

In this section, we conduct comprehensive experiments to compare the results of our proposal with those of the baselines. Furthermore, we investigate the performance of the hybrid QAC model under various settings, for example, WMA, SMA, and changing sliding windows for burst detection.

5.1. Periodicity-Based QAC Model Performance. To answer (RQ1), we report the MRR scores of QAC rankings generated by MPC-ALL, MPC-W, and our periodicity-based QAC model P-QAC at various lengths of prefix ($\#p$) in Table 1.

It can be seen from Table 1 that, for the MPC-W model, it performs best using a 28-day time window for all cases except that $\#p = 1$, where MPC-W with a 7-day time window achieves the highest MRR score. In addition, we observe that MPC-W generally beats MPC-ALL as for most cases MPC-W can receive higher MRR scores than MPC-ALL except the case that at $\#p = 3$ MPC-ALL presents a slight improvement

Table 1: Performance of QAC models in terms of MRR on the AOL dataset for prefix p consisting of 1–5 characters. The best results of baselines and of all models at each row are italic and bold, respectively.

$\#p$	MPC-ALL	MPC-W					P-QAC
		2 days	4 days	7 days	14 days	28 days	
1	0.1097	0.0868	0.1066	*0.1113*	0.1110	0.1109	**0.1115**
2	0.2012	0.1780	0.1894	0.1952	0.1985	*0.2029*	**0.2059**
3	*0.3211*	0.2653	0.2792	0.3005	0.3114	0.3195	**0.3287**
4	0.4155	0.3625	0.3918	0.4028	0.4126	*0.4214*	**0.4356**
5	0.4951	0.4370	0.4691	0.4800	0.4913	*0.5028*	**0.5163**

against MPC-W in terms of MRR. It can be explained by the fact that, by taking the query recency into account, MPC-W is able to provide users with a list of newly popular query completions that are not consistently popular.

In addition, P-QAC outperforms the baselines for all prefix lengths in terms of MRR. In detail, for prefix lengths of 1–3 characters, P-QAC shows relatively small improvements (under 1%) over the best performer of MPC-W; however, as the prefix length gets longer, the improvement of P-QAC against the best MPC-W (with a 28-day time window) is enlarged. For instance, the MRR improvements come to near 1.5% at $\#p = 4$ and 1.4% at $\#p = 5$, respectively. This implies that our proposal using the predicted query popularity based on the periodicity can generate better QAC rankings than the baselines. Consequently, the cyclic pattern of query history popularity serves as a reliable ranking signal which can be exploited to obtain better QAC rankings.

5.2. Hybrid QAC Model Performance. In this section, we turn to (RQ2) and examine the improvements of our hybrid QAC models λ-H-WMA and λ^*-H-WMA against P-QAC. Both the two hybrid models work under the setting of a 7-day sliding window when detecting the burst. The evaluation results of three different ranking models are included in Figure 2.

In general, the empirical results indicate that the hybrid QAC models λ-H-WMA and λ^*-H-WMA can outperform P-QAC by a large margin for all prefix lengths. Specifically, two most significant improvements of the hybrid models against P-QAC are produced by λ^*-H-WMA, which reports an MRR improvement up to 2.5% and 2.2% at $\#p = 3$ and $\#p = 5$, respectively. It could be due to the fact that a large amount of queries exhibits burst trends in their popularity. The outcomes verify our proposal that incorporates the burst trend into the periodicity-based QAC model can further exploit temporal patterns of query popularity for better ranking performance. Compared to P-QAC, by combining both the long-term periodicity and the short-term burst, λ-H-WMA and λ^*-H-WMA can correctly capture the signal of time-sensitivity in predicting query's future popularity and provide reasonable ranking lists of query completions to searchers.

In addition, as to the two hybrid QAC models, λ^*-H-WMA beats λ-H-WMA for all prefix lengths. Generally, λ^*-H-WMA is more effective for longer prefixes than λ-H-WMA. Although the improvement of λ^*-H-WMA over λ-H-WMA is relatively small with 0.34% at $\#p = 2$, it is further expanded to more than twice of that with 0.84% under $\#p = 5$. The comparing results confirm our proposal that the flexible value λ^* can capture the variations in burst amplitudes of query popularity and further improve QAC rankings.

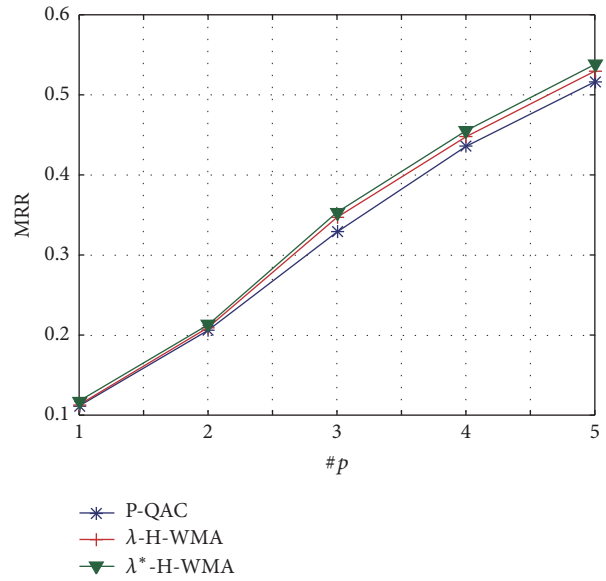

Figure 2: Absolute MRR scores of P-QAC, λ-H-WMA, and λ^*-H-WMA at prefix length $\#p = 1$ to 5 on the AOL query log. Both λ-H-WMA and λ^*-H-WMA use a sliding window of 7 days.

5.3. Impact of Decay Function. To answer (RQ3), we compare the performance of λ-H-SMA and λ-H-WMA using a 7-day sliding window at prefix length $\#p = 1$ to 5. We report the MRR scores in Table 2.

Clearly, from Table 2, we find that λ-H-WMA outperforms λ-H-SMA at all prefix lengths with slight improvements in terms of MRR. In particular, for cases $\#p = 2$ and $\#p = 4$, relatively obvious improvements in terms of MRR are observed when comparing λ-H-WMA against λ-H-SMA, reporting near 0.21% and 0.32% improvements, respectively. It indicates that λ-H-WMA can predict the user's intended query more accurately than λ-H-SMA.

In addition, the experimental outcomes reported in Table 2 infer that when detecting the burst of query

Table 2: MRR scores of λ-H-SMA and λ-H-WMA, tested at prefix length $\#p = 1$ to 5 on the AOL dataset.

$\#p$	λ-H-SMA	λ-H-WMA
1	0.1126	0.1130
2	0.2073	0.2094
3	0.3456	0.3470
4	0.4444	0.4476
5	0.5277	0.5296

Table 3: Ratios of unique queries with bursts identified by λ-H-WMA using different lengths of sliding window.

Window length	2 days	4 days	7 days	14 days	28 days
Ratio	44.3%	43.5%	43.6%	44.1%	52.9%

popularity, assigning flexible weights to previous observations according to the time interval is better than using uniform weights. It is consistent with the findings observed in temporal information retrieval that the more recent the data is, the more reliable it is in predicting future trend [14, 22]. Thus, the weights of past observations are monotonically increasing as the time interval shortens. That is, the recent data contributes more than the remote one. Accordingly, we conclude that introducing a decay function to the burst detection step in our hybrid QAC model helps improve the overall QAC performance.

5.4. Impact of Sliding Window Length. In order to answer (RQ4), we first vary the length of sliding window in burst detection and then examine the impacts on QAC performance. Following [1], we employ 5 different sliding windows with a range of 2 to 28 days and report the ratios of queries with bursts in Table 3.

We can see from Table 3 that the λ-H-WMA model with a 28-day sliding window identifies the largest amount of queries with bursts. In contrast, when using a shorter sliding window, for example, 4 or 7 days, it detects less bursty queries. Clearly, the length of sliding window does affect the burst detection based on a query's past popularity.

Next, we examine the performance of λ-H-WMA with 5 different sliding windows for prefixes consisting of 1–5 characters and report the absolute MRR scores in Figure 3.

As shown in Figure 3, in general, the λ-H-WMA model performs best under the setting of a 7-day sliding window except the case that a marginal drop of 0.45% in terms of MRR is observed at $\#p = 1$ when comparing to its performance under the setting of a 4-day sliding window. It is also worth noting that the λ-H-WMA model with a 28-day sliding window results in the lowest MRR values for all prefix lengths. In detail, for $\#p = 3$, 4, and 5, the λ-H-WMA model with a 7-day sliding window leads to an MRR increase of up to 1.4%, 1.7%, and 1.3% over the performance under the setting of a 28-day sliding window, respectively. This indicates that, as the prefix becomes longer, λ-H-WMA with a sliding window of 7 days consistently performs better than that under other settings.

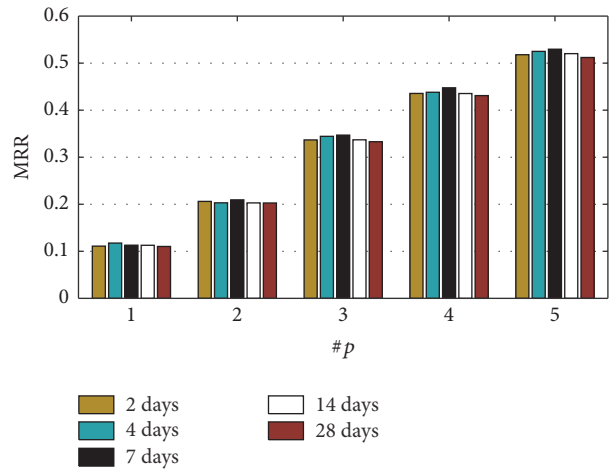

Figure 3: MRR values of λ-H-WMA with sliding window lengths of 2 to 28 days and query prefix p lengths of 1–5 characters.

Interestingly, as shown in Table 3, the λ-H-WMA model with a 28-day sliding window leads to the largest number of bursty queries; however, the corresponding QAC performance of λ-H-WMA is not as efficient as that with other settings of sliding windows. In contrast, λ-H-WMA achieves the highest MRR score with a 7-day sliding window but it detects less bursty queries. It could be explained by the fact that the majority of query bursts in the AOL dataset are short-term bursts and sliding windows of 14 or 28 days will depress the short-term variations. However, too short sliding windows, for example, 2 or 4 days, will exaggerate the short-term fluctuations and introduce noise in burst detection. Therefore, bursty queries can be efficiently identified under the setting of a 7-day sliding window, which helps to improve the QAC performance.

6. Conclusion

In this paper, we present a novel approach to address the problem of query autocompletion (QAC). In detail, we consider the periodicity and the burst trend to predict a query's future popularity for ranking query completions, which is achieved by the Discrete Fourier Transform and Weighted Moving Average, respectively. We combine these two temporal patterns by proposing a hybrid QAC model. Finally, we verify the effectiveness of our proposal on the AOL query log, showing remarkable improvements in terms of Mean Reciprocal Rank over typical QAC baselines.

As future work, investigating other temporal patterns of query popularity may improve the quality of query completions. In addition, user specific information such as gender, age, and click-through data can be incorporated into the current hybrid QAC model to generate a personalized QAC list to a specific user.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is partially supported by the National Advanced Research Project under no. 6141B08010101.

References

[1] S. Whiting and J. M. Jose, "Recent and robust query auto-completion," in *Proceedings of the 23rd International Conference on World Wide Web*, pp. 971–981, Seoul, Republic of Korea, April 2014.

[2] Z. Bar-Yossef and N. Kraus, "Context-sensitive query auto-completion," in *Proceedings of the 20th International Conference on World Wide Web (WWW '11)*, pp. 107–116, ACM, Hyderabad, India, April 2011.

[3] A. Strizhevskaya, A. Baytin, I. Galinskaya, and P. Serdyukov, "Actualization of query suggestions using query logs," in *Proceedings of the 21st Annual Conference on World Wide Web (WWW '12)*, pp. 611–612, Lyon, France, April 2012.

[4] M. Shokouhi and K. Radinsky, "Time-sensitive query auto-completion," in *Proceedings of the 35th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 601–610, Portland, Ore, USA, August 2012.

[5] M. Shokouhi, "Learning to personalize query auto-completion," in *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '13)*, pp. 103–112, Dublin, Ireland, August 2013.

[6] S. Bhatia, D. Majumdar, and P. Mitra, "Query suggestions in the absence of query logs," in *Proceedings of the 34th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 795–804, Beijing, China, July 2011.

[7] S. Chaudhuri and R. Kaushik, "Extending autocompletion to tolerate errors," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 707–718, Providence, RI, USA, July 2009.

[8] B.-J. Hsu and G. Ottaviano, "Space-efficient data structures for top-κ completion," in *Proceedings of the 22nd International Conference on World Wide Web*, pp. 583–594, Rio de Janeiro, Brazil, May 2013.

[9] X. Li and W. B. Croft, "Time-based language models," in *Proceedings of the 12th ACM International Conference on Information and Knowledge Management (CIKM '03)*, pp. 469–475, New Orleans, La, USA, November 2003.

[10] F. Diaz and R. Jones, "Using temporal profiles of queries for precision prediction," in *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 18–24, Sheffield, UK, July 2004.

[11] A. Kulkarni, J. Teevan, K. M. Svore, and S. T. Dumais, "Understanding temporal query dynamics," in *Proceedings of the 4th ACM International Conference on Web Search and Data Mining*, pp. 167–176, Hong Kong, February 2011.

[12] J. L. Elsas and S. T. Dumais, "Leveraging temporal dynamics of document content in relevance ranking," in *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining*, pp. 1–10, New York, NY, USA, February 2010.

[13] K. Berberich, S. Bedathur, O. Alonso, and G. Weikum, "A language modeling approach for temporal information needs," in *Proceedings of the 32nd European Conference on Advances in Information Retrieval*, pp. 13–25, Milton Keynes, UK, March 2010.

[14] N. Kanhabua and K. Nørvåg, "Determining time of queries for re-ranking search results," in *Proceedings of the 14th European Conference on Research and Advanced Technology for Digital Libraries (ECDL '10)*, pp. 261–272, Glasgow, UK, 2010.

[15] R. Campos, G. Dias, A. M. Jorge, and C. Nunes, "Enriching temporal query understanding through date identification: how to tag implicit temporal queries?" in *Proceedings of the 2nd Temporal Web Analytics Workshop (TempWeb '12)*, pp. 41–48, Lyon, France, April 2012.

[16] M. Shokouhi, "Detecting seasonal queries by time-series analysis," in *Proceedings of the 34th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1171–1172, Beijing, China, July 2011.

[17] M. Vlachos, C. Meek, Z. Vagena, and D. Gunopulos, "Identifying similarities, periodicities and bursts for online search queries," in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '04)*, pp. 131–142, Paris, France, June 2004.

[18] Z. Li and J. Han, *Data Mining and Knowledge Discovery for Big Data: Methodologies, Challenge and Opportunities*, Springer, Berlin, Germany, 2014.

[19] M. Vlachos, P. Yu, and V. Castelli, "On periodicity detection and structural periodic similarity," in *Proceedings of the 5th SIAM International Conference on Data Mining (SDM '05)*, pp. 449–460, Newport Beach, Calif, USA, April 2005.

[20] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition, 2009.

[21] M.-H. Peetz, E. Meij, and M. de Rijke, "Using temporal bursts for query modeling," *Information Retrieval*, vol. 17, no. 1, pp. 74–108, 2014.

[22] P. N. Bennett, R. W. White, W. Chu et al., "Modeling the impact of short- and long-term behavior on search personalization," in *Proceedings of the 35th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 185–194, Portland, Ore, USA, August 2012.

[23] G. Pass, A. Chowdhury, and C. Torgeson, "A picture of search," in *Proceedings of the 1st International Conference on Scalable Information Systems*, Hong Kong, June 2006.

[24] J. Gama, I. Zliobaite, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Computing Surveys*, vol. 46, no. 4, article 44, 2014.