This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2736064, IEEE Access

Tianji: Implementation of an Efficient Tracking Engine in Mobile Internet Era

Jin-Yuan Chen, Hai-Tao Zheng*, Xi Xiao, Arun Kumar Sangaiah, Yong Jiang, Cong-Zhi Zhao

***Abstract*—In the mobile Internet era, users tend to get interested information in a continuous manner rather than get one-time result through search engines. The traditional link-based ranking algorithms typically return the relevant "popular" web pages. The current most important web pages are ranked lower than these pages. Besides, most of the results are even the same when the user submits the same query some days later. In this work, we have described a novel service: Tracking Engine. The tracking engine allows the users to input and save queries, displays time-sensitive information for the users and notifies users when new relevant information appears. To the best of our knowledge, this is the first solution seen in such service in mobile internet era. First, our tracking engine called Tianji crawls the web pages on the basis of the time priority, and constructs a new index structure which enables enables every web page match to the related keywords timely. Then, we develop a ranking model based on the correlation between time and importance. The experimental results show that the ranking model of Tianji has better performance than existing time-sensitive ranking methods in terms of timeliness and relevance.**

***Index Terms*—Tracking Engine, keyword search, temporal search, Mobile Internet**

I. Introduction

With the rapid development of mobile Internet [1], [2], the wide variety of mobile users want to find information they are interested in any time and place. Since the mobile screen is really small, it might be more difficult to type queries in mobiles than personal computers. A better provision for mobile users would be a service that records user queries in the server, and notifies them when new relevant information is found.

Traditional search engines [3] have limited capability and are not the optimum choices for mobile users. Firstly, traditional crawlers such as [4], [5] is focused on downloading the entire web sites, not the newly appeared web pages. Thus some newly published web pages are discovered several days later. These web pages are much more important than the old ones. Secondly, traditional search engines cannot track relevant information for users. Users have to search the same query again when they want to find new information. When the user searches the same query twice, the ranking result is similar as the first time, without considering the user history. Therefore, it is hard to locate new information in traditional search engines. Thirdly, temporal information is ignored in traditional search engines. The ranking results are based on relevance and authority [6]. In this way, many relevant but overdue results are placed before timely results. For example, when searching "high tide" in Google on May 26, 2016, no results relating to Hawaii's high tide alert is found on the first page. The alert was published on May 24, and was highly concerned (the 16th query on Google Trends). Many predecessors have considered the third problem and proposed works about recency sensitive searching models [7]–[18]. Besides, Google news [1] and Baidu news [2] are also published to solve this problem. These works consider the influence of time when ranking web pages but neglect the user's time perception. According to Cheng luo et al. [19] the Vierordts law [20] has been consistently observed in Web search environment. Users are more sensitive to the recent information, and less for the outdated information. For example, we want to know the events happened today as soon as possible. But for the events happened one year ago, we are more interested in the events themselves not the time they happened. Thus, none of the three problems have been completely solved till now.

In order to solve these three problems in a better way, we propose a novel web service Tracking Engine. The tracking engine allows users to input and save queries, displays recency sensitive information for the users and notifies users when new relevant information appears. In this work, we have developed a tracking engine named Tianji especially for mobile users. To the best of our knowledge, this is the first solution of such service for mobile internet users.

Our proposed tracking engine Tianji records the user queries when required and informs users when new relevant information is found by the tracking engine. First, we developed a web crawler which prefers to download new web pages. Our web crawler learns the authorities and the update frequencies of the web sites, and downloads web pages on the basis of the publish time. Following this, we built a tracking model that stores and indexes the user strategies (similar with queries in search engines). Each new web page is searched over existing strategies to find corresponding users. These users are informed of the new web page if necessary. Eventually, we

Jin-Yuan Chen, Hai-Tao Zheng, Xi Xiao and Yong Jiang are with the Graduate School at Shenzhen, Tsinghua University, Shenzhen, Guangdong, CN 518055.

Arun Kumar Sangaiah is with the School of Computing Science and Engineering, VIT University, Vellore-632014, India.

Cong-Zhi Zhao is with the Giiso Information Technology Co., Ltd Shenzhen, Guangdong, China.

This research is supported by National Natural Science Foundation of China (Grant No. 61375054), Natural Science Foundation of Guangdong Province Grant No. 2014A030313745Basic Scientific Research Program of Shenzhen City Grant No. JCYJ20160331184440545), and Cross fund of Graduate School at Shenzhen, Tsinghua University (Grant No. JC20140001).

[1]https://news.google.com/

[2]https://news.baidu.com/

developed a time-based ranking model to find the important and timely web pages. The ranking model is based on a reciprocal decay function. An important characteristic of this ranking model is that the ranking result of the same query changes when the search time changes. The main contributions of this proposed work are listed as follows:

1) We are the first to propose and develop the new type of web service Tracking Engine. Tracking Engine aims to help users obtain the information of their interest in a continuous manner.
2) We present an inverse search method, ie. identify relevant user queries for input documents.
3) We have developed a ranking model which utilizes a reciprocal decay function to evaluate the timeliness requirements of users.

Our tracking engine Tianji has served more than 5 million users in China. The experiments we conducted prove the efficiency of our design. The reciprocal decay function performs better than all the baseline methods in terms of precision and NDCG.

The paper is organized as follows: The next section carries a review of the related work in a temporal searching area. Section 3 elaborates the features of tracking engine Tianji. Section 4 shows the evaluation of our method. The last section presents the conclusion and the scope for future work.

II. Related work

To the best of our knowledge, this is the first paper introducing Tracking Engine for mobile users. The searching situation in a tracking engine is similar to the recency sensitive searching models. Thus we introduce the previous works relating to recency sensitive searching in this section. Surveys of temporal information retrieval including recency sensitive searching models can be found in [21], [22].

Li and Croft [7] were the first researchers that proposed a time-based ranking model in information retrieval tasks. They evaluated the search result distribution, and showed that a numerous number of queries are time sensitive. Based on the observation, they utilized a time-based priority to incorporate document time into the relevance measurement. So the probability given q for a document d is: $p(d|q) \propto p(q|d)p(d|T_d)$, where T_d is the document creation/publication time. Time sensitive queries can be divided into two parts: recency query and time period query. Time period queries prefer documents relating to particular period. They estimate such queries with a normal distribution priority function. Recency queries prefer documents published recently. They utilize an exponential priority function to estimate these queries.

Efron and Golovchinsky [17] have proposed an extension of the query likelihood model considering both document publication time and the relationship between publication time and the query. They have also proposed a temporally informed smoothing method which enables aggressive smoothing of the older documents. However, their methods can only boost the queries whose relevant document volume increases over time. To overcome this limitation, Cheng et al. [18] pointed out that not only recency queries but also queries that appear periodically prefer recent documents, such as "credit card overdraft fees". These queries are called "timely queries". They have proposed a method estimating the query timeliness requirements based on the change of query results' term distribution. They use the estimating result to setup the parameter λ in the exponential distribution method [7]. Their method works well in traditional searching tasks. But, for tracking engines while the timeliness requirements are more rigorous, the estimating result is not that good.

Dong et al. [12] have proposed a learning-to-rank based method. They trained a ranking model based on timestamp features, linktime features, webbuzz features and page classification features. A user query is first classified by the recency sensitive query classification. Recency queries alone are passed on to the ranking model. Subsequently, Inagaki et al. [13] proposed a set of novel click features to improve machine learning methods for recency ranking. Dong et al. [14] have proposed a method which uses twitter blogging data to detect fresh web pages and to compute novel and effective features for these pages.

Dai et al. [15] noticed that the failure of recognizing temporal aspects of user queries can have a negative effect on the search results. To minimize the negatively affect caused by classification error, they proposed a method combining both recency ranking and regular ranking together. The method is based on DAC [23] model, combining different ranking model results based on the probability the query belonging to the model. Styskin et al. [16] have proposed a diversification method for recency ranking. They combined recency results with ordinary results according to the recency query classification result.

Berberich et al. [8] have proposed two link analysis based approaches to estimate the freshness of web pages: T-Rank Light and T-Rank. Their method takes both link-structure and freshness of the web page into consideration. Similarly, Cho et al. [9] have proposed a new ranking metric to improve the temporal bias problem [24] of PageRank [6]. PageRank fails to promote newly created web pages as they have less incoming links. After that, in 2010, Dai and Davison [10] proposed a method T-Fresh which considers the variation of page and in-link freshness together. These methods can evaluate the authorities of new web pages better, but the requirements of query timeliness are not considered during the search.

Zhang et al. [11] studied the implicit time requirements in queries. For example, when a user queries "sigir" in 2017, it probably means "sigir 2017" but not "sigir 2015". These queries are named as year qualified queries (YQQs). They proposed a method to identify YQQ and boost the results which match the implicit year.

Our work focuses on user strategies in tracking engine which also favor recently published documents. User strategies are the queries which satisfy user interests, for example: "Kobe Bryant", "Lakers", "Lady Gaga", Nasdaq Stocks and so on. The users prefer to know the most recent news of such queries. The older news are probably already read by users. Thus user strategies are always recency sensitive. Unlike recency sensitive queries, relevant results of strategies exist all the time, every week or even every day. Unlike "timely queries",

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2736064, IEEE Access

the term distributions of user strategies change little over time and are not correlative with the users' timeliness requirements.

III. Tracking Engine

In this paper, we provide the description of a new type of web service: Tracking Engine. Tracking Engine aims to help users keep sustained attention on the information they are interested in or concerned. Users input their strategies in the tracking engine. The tracking engine find relevant information for the users on a continuous basis. In this case, users can read relevant information continuously without redundant searching. Definition 1 shows the definition of Tracking Engine.

Definition 1: Given a user strategy T, a time slot $S : (t_1, t_2)$ and the document set D, Tracking Engine aims to find a subset R of D:

$$\arg\max_{M} P(R|T, S, D)$$

where t_1 is the time corresponding user previously read the strategy, t_2 is the time this user read the strategy (normally current), M is the Tracking Engine being designed.

The user strategy T is a special kind of query, which is typed to search for information. Unlike queries in search engines, user strategies are stored and indexed by the tracking engine. After the user strategy is stored, new relevant information crawled by the web crawler will be automatically added to the user's profile. The user will receive a notification (if subscribed) relating to the new information. Thus when the user visits our service again, the new information will be automatically ranked and displayed for the user.

In practice, we developed a Tracking Engine named Tianji. Tianji consists of three parts, the Web Crawler, the Tracking Model and the Ranking Model. Figure 1 shows the framework of Tianji. Web crawler downloads new web pages from the

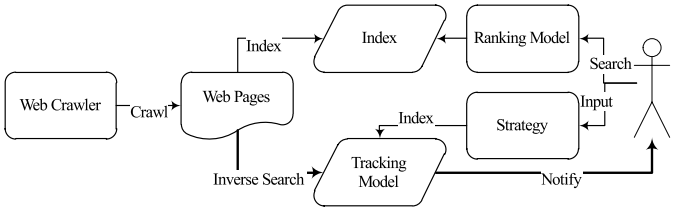

Fig. 1. The framework of Tracking Engine Tianji

Internet. New web pages are indexed by the indexer for searching. At the same time, the new web pages are sent to Tracking Model. Tracking Model finds the relevant user strategies for the downloaded web pages. The corresponding users are notified by the Tracking Model, if necessary. On the user side, users save their strategies in the server. The Tracking Model indexes the user strategies for tracking. Users can directly view newly appeared pages simply by clicking the strategy they saved. The search request is submitted to Ranking Model which sorts results considering both content relevancy and timeliness requirement. Besides, if the strategy has been read earlier, the Ranking Model ranks unread web pages in front of the other results.

The main differences between Search Engines and Tracking Engines include three aspects. First, Tracking Engines notify users when new information is found, while Search Engines do not. Secondly, Tracking Engines record users' read log, and put unread information on top, while Search Engines record users' log for behavior analysis. Thirdly, Tracking Engines take temporal information into consideration, while most Search Engines consider the contents only.

A. Web Crawler

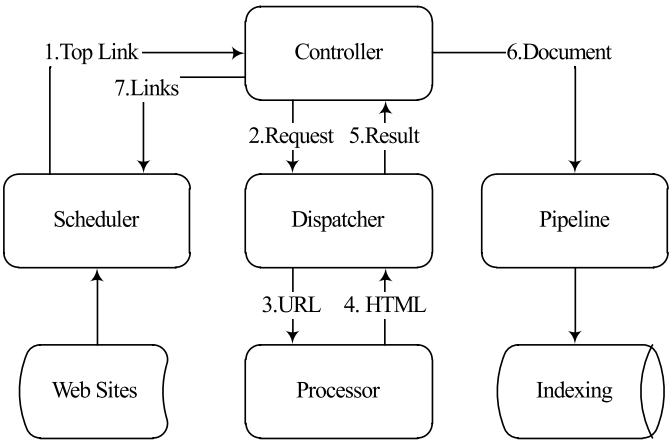

Fig. 2. The structure of our web crawler

Figure 2 shows the structure of our web crawler. The web crawler consists of five components: Scheduler, Controller, Dispatcher, Processor and Pipeline. Scheduler manages the download status and the queue of the undownloaded links. When a new link is added to the Scheduler, it calculates the link's priority and puts it in the right space. Besides, when a web page is downloaded, the Scheduler updates the corresponding links' priority and resorts the link queue. Controller controls the entire system. It obtains a new URL from the Scheduler, sends the URL request to the Dispatcher, receives downloaded results from Dispatcher, adds new links to the Scheduler and instructs pipelines to process the document. Dispatcher dispatches URL requests to the processors. And Processors download the web pages. The Dispatcher and the Processors are deployed in different machines. Therefore, the web crawler can use different networks to enhance download speed. At last, Pipeline indexes the downloaded documents.

Since our web crawler is developed for tracking engine, newly published information is more important than old information. In practice, we noticed that most newly published web pages with high quality appear on the home pages or the pages directly linked from the home page. Therefore, our scheduler ranks links on the basis of their landing page. We assign a priority value for each link; the home pages are assigned with a constant priority. Other links' priorities are assigned according to equation (1).

$$pri(l) = \max_{pg \in PG} pri(pg) - 1 \tag{1}$$

where l is target link, PG is the web pages containing link l. In our web crawler, we modify the corresponding links' priorities when a new web page is downloaded. And the home pages are visited periodically.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2736064, IEEE Access

B. Tracking Model

The user strategies are stored in the Tracking Engine. The Tracking Engine needs to find relevant user strategies for each new web page. Since our web crawler downloads more than 500 pages every minute, the efficiency of finding relevant strategies is extremely important. In Tianji, each strategy consists of query terms and some other restrictions. The main problem is how to find strategies that match with the web page in term level. Figure 3 shows the structure of our tracking model. The left side of Figure 3 is the index phase of the

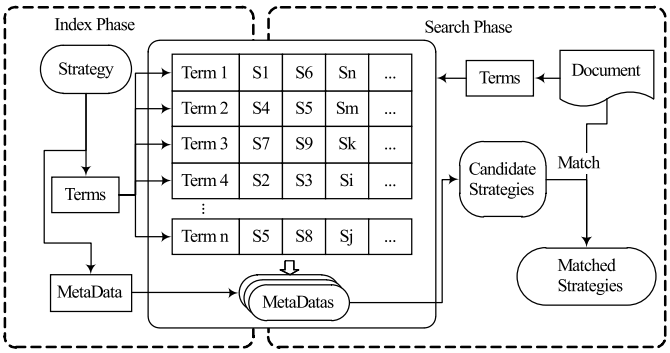

Fig. 3. The structure of our Tracking Model

strategies. The right side is the search phase for new web pages.

In the index phase, we analyze the user strategies, and segregate them into terms and metadatas. Our index maintains an inverse index for all the strategies. Each term is followed by a strategy list, i.e. $S1$ - Sn in the figure. The metadata is stored directly in the index. Metadata includes the number of terms, the phrase queries and some other restrictions.

In the search phase, each web page is separated into terms. These terms are used to find candidate strategies from the inverted index. We check the term number in metadata and the matched term number for each candidate strategies. If the two numbers are equal, we can conclude that the strategy matches with the web page in term level. After that, we check the phrase queries and the other restrictions. If all of them match, the web page is relevant to the strategy.

C. Ranking Model

Search engines rank documents according the relevance between the query and the documents. A classic function calculates the relevance score is the cosine similarity between the query vector and the document vector. In this work, we use Lucene [25] to rank web pages. The score function of Lucene is equivalent to equation (2).

$$score(q,d) = \frac{\|t \in q \& t \in d\|}{\sqrt{length\ of\ d} \times \|t \in q\|} \sum_{t \in q} (tf(t,d) \times idf(t)^2) \tag{2}$$

where q is the query, d is the document needs to be scored, t is a term, $idf(t)$ is the inverse document frequency [26] of t and $tf(t,d)$ is the times of t appears in d.

While ranking, the main difference between tracking engine and search engine is tracking engines must take temporal information into consideration. The strategies in tracking engine are always time sensitive. Older information should be devaluated and newer information is more valuable for users.

Similar with previous works [7], [18], we utilize the probability dependent or document date $P(d|T_d)$ to evaluate the freshness level of documents. The final score of a document is shown as follows.

$$time\_score(q,d) = score(q,d) \times P(d|T_d) \tag{3}$$

In order to evaluate the time priority as close as humans do, exponential decay function is used in [7], [18].

$$P(d|T_d) = P_{exp}(T_d) = \lambda e^{-\lambda(t_s - T_d)} \propto e^{-\lambda(T_s - T_d)} \tag{4}$$

where λ is the rate parameter in exponential distribution, t_s is the time the user submits the query, T_d is the document's publish time. Thus older documents are scored less than newer ones.

Cheng luo et al. [19] investigates users' time perception during web search, and finds the Vierordts law [20] is consistently observed in Web search environment. I.e. shorter intervals tend to be overestimated while longer intervals tend to be underestimated. For example, events happened yesterday are very fresh for users, while the events happened a week ago is not so fresh. But for events happened a year ago and events happened 13 months ago, they sound similar in fresh level. Based on this founding, we deem that exponential decay is not the best choice for evaluating document priority. Exponential decay cannot satisfy this feature. The decay ratio is stable over time. Equation (5) shows the decay ration of exponential decay. From this equation, we can find the decay ration is only determined by λ.

$$\begin{aligned} \frac{P_{exp}(T_{d_2})}{P_{exp}(T_{d_1})} &= \frac{e^{-\lambda(t_s - T_{d_2})}}{e^{-\lambda(t_s - T_{d_1})}} \\ &= e^{-\lambda(T_{d_1} - T_{d_2})} \end{aligned} \tag{5}$$

Intuitively, we can compare the $time\_score$ of two documents d_1 and d_2. From equation (6) we can find the comparison result is determined by the documents content and the rate parameter λ. This means the ranking result is always the same.

$$\frac{time\_score(q,d_2)}{time\_score(q,d_1)} = \frac{score(q,d_2)}{score(q,d_1)} \times e^{-\lambda(T_{d_1} - T_{d_2})} \tag{6}$$

But in temporal search domain, the ranking result should be changed while the search time changes. For example, assuming a Hong Kong user searches "Rio Olympic Games" in August 22th, 2016, the news about "Chinese Olympic winners visiting Hong Kong" is the most important news. But if this user searches the same query in 2017, this news is no longer important, but the news about "How many medals Chinese win" is still important. Thus the ranking result between the two news changes. To solve this problem, we propose the reciprocal decay function.

$$P_{rec}(T_d) = \frac{\beta}{\beta + t_s - T_d} \tag{7}$$

In this way, the time priorities comparison between d_1 and d_2 changes to equation (8). The $time\_score$ comparison is equation (9).

$$\frac{P_{rec}(T_{d_2})}{P_{rec}(T_{d_1})} = \frac{\beta + t_s - T_{d_1}}{\beta + t_s - T_{d_2}} \tag{8}$$

$$\frac{time\_score(q, d_2)}{time\_score(q, d_1)} = \frac{score(q, d_2)}{score(q, d_1)} \times \frac{\beta + t_s - T_{d_1}}{\beta + t_s - T_{d_2}} \tag{9}$$

Obviously, the ranking result between d_1 and d_2 changes while the search time changes. From equation (8), we can see T_{d_1}, T_{d_2} is constant. Thus when the searching time t_s increases, $\frac{P_{rec}(T_{d_2})}{P_{rec}(T_{d_1})}$ also increases. The term level similarity becomes more important than the temporal bias.

IV. Experimental Evaluation

In this section, we exhibit the experiment results of our work. The next subsection introduces the datasets setup, the baseline methods and the effectiveness matrices used in our experiment. After that, we propose the experiment results in four parts: the parameter setup, the effectiveness comparation, the different ranking results over time and the web crawler efficiency.

A. Experiment Setup

Datasets. Our experiment is built on a real system which serves more than 5 million users in China. Our web crawler downloads more than 10 thousand different pages every hour. When the experiment is conducted, we have more than 100 million documents in our database. We have selected 54 top queries according "Baidu top Searches" [3]. These queries include recency events, famous persons, popular games, famous competitions, stocks, etc.

After the queries are collected, each query is sent to the Tianji with different decay functions and different parameters. The top 10 diversified search results are returned for each query. Three judges were asked to tag the search results with the following four grades:

Irrelevant: The search result is not relevant with the query specified.

Relevant but Old (Old): The search result is relevant but it's too old and is of no use now.

Relevant and Useful (Useful): The search result is relevant and useful, but it's not the newest information.

Relevant and Newest (Newest): The newest information for the query.

Tag results agreed to two or more judgments are selected as the correct answer. In the event of disagreement between the judgments, a final judger is asked to select one choice from them. Table I shows the statistic of our dataset.

Baselines. Four baseline methods are used to evaluate the effectiveness of our method.

Baseline 1: The first base line method (LUCENE) is the traditional ranking system provided by Lucene [25]. The score function is proposed in equation (2).

Baseline 2: The second base line method (SORT) is the sorting of results by publish time.

[3]http://top.baidu.com/boards

TABLE I
Statistic of the Dataset

Total Docs	Query	Results per query	Total Tags
107,638,593	54	49.6	2680
Irrelevants	**Olds**	**Usefuls**	**Newests**
759	893	571	457

Baseline 3: The third base line method (EXP) utilizes the exponential decay function in equation (5), which is proposed by [7].

Baseline 4: The fourth base line method is TDC, proposed by [18]. They also use exponential decay function. But they calculate the parameter λ by terms distribution change (TDC). Given a query Q, TDC is calculated as follows:

$$TDC(Q) = \frac{1}{n-1} \sum_{i=1}^{n-1} KL(LM(T_i), LM(T_{i+1})) \tag{10}$$

Where T_i is the ith time slot, $LM(T_i)$ refers to the term distribution of the search results in T_i, KL denotes the Kullback-Leibler (KL) divergence [27]. In our experiment, we set the length of each time slot to 30 days since our crawler starts from 2015. And the search results of the past one year (eg. 12 time slots) are used for calculating the TDC.

Effectiveness Metrics. In our experiment, we applied two widely used relevance metrics for evaluation of the effectiveness of the methods. The first one is precision. The precision value of top n results is calculated as follows:

$$P@n = \frac{relevant\ results}{n} \tag{11}$$

Since we have four relevant grades, we treat Useful and Newest results as relevant results while calculating precision. The results tagged as "Relevant but Old" is processed as irrelevant because it's useless for the users.

The second one is NDCG (Normalized Discounted Cumulative Gain) [28]. NDCG is calculated as follows:

$$DCG@n = \sum_{i=1}^{n} \frac{2^{rel_i} - 1}{\log_2(i+1)} \tag{12}$$

$$NDCG@n = \frac{DCG@n}{IDCG@n} \tag{13}$$

Where rel_i is the relevance score of the ith result. The results tagged with "Irrelevant" are scored with 0, "Relevant and Old"s are 1, "Useful"s are 2 and "Newest"s are 3. $IDCG@n$ is the ideal $DCG@n$, i.e. the maximum possible DCG value up to the ranking position n.

B. Experiment Results

In this subsection, we present various experiments to demonstrate the effectiveness of our framework. In the Parameter Setup, we find the parameter in REC is easier to determine, and is similar to human's cognition. In Effectiveness Comparation we prove that REC is the best decay function during tracking. In Ranking Results over Time, we testify the decay speed of REC goes down for older information. And in Web Crawler Efficiency, we demonstrate that most new web pages are downloaded in two hours.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2736064, IEEE Access

1) Parameter Setup: We need to set the parameters in three methods, i.e., λ in EXP, β in REC (the reciprocal decay function) and α in TDC method. The effectiveness of λ and β values cannot be inferred intuitively. Therefore, we utilize another parameter p_{30} instead. p_{30} is the decay value of a document published 30 days ago for the target method. For EXP, we can calculate λ given p_{30} as: $\lambda = -\ln p_{30}/30$. And for REC, β is $30 \times p_{30}/(1 - p_{30})$. Figures 4 and 5 show the NDCG@5 measurement of different parameters in TDC, EXP and REC. These figures help finding the performance for

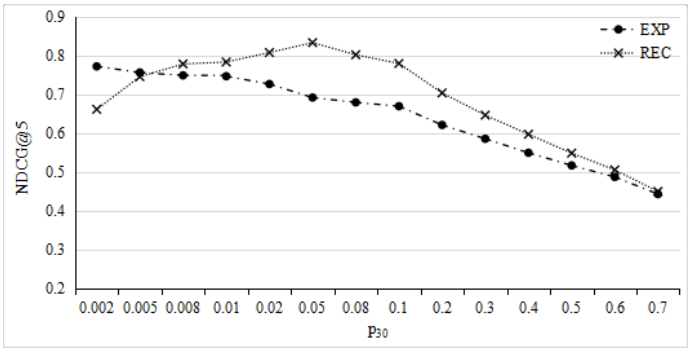

Fig. 4. Parameter p_{30} of EXP, REC

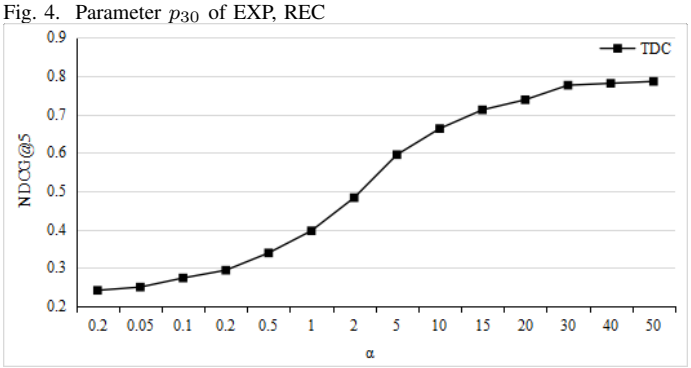

Fig. 5. Parameter α of TDC

REC is the best when $p_{30} = 0.05$. This parameter is similar to our intuition, the importance of documents published one month ago is about 1/20 of the ones published currently. For EXP, p_{30} is set to 0.002 in the rest of our experiment. For the EXP method, it is possible there exists a better p_{30}, but the precision of p_{30} is high enough. Checking values such as 0.001, 0.0005 is unfair for the other methods, etc. we did not check the values between 0.05 and 0.08. We set $\alpha = 50$ for TDC because the curve is almost flat when $\alpha = 50$.

From this experiment, we can conclude that the parameter of REC is easier to determine than that of EXP and TDC. This is because reciprocal decay function is similar to humans cognition. The old messages are memorized similar by people, while fresh messages are memorized significantly different.

2) Effectiveness Comparation: Figure 6 shows the effectiveness of Lucene, SORT, EXP, TDC and REC in terms of precisions and NDCG. From this figure we can find REC performs the best among the five methods. The precisions of P@1, P@3, P@5 are 94.44%, 95.06% and 94.07% respectively. The NDCGs of NDCG@1, NDCG@3, NDCG@5 and NDCG@10

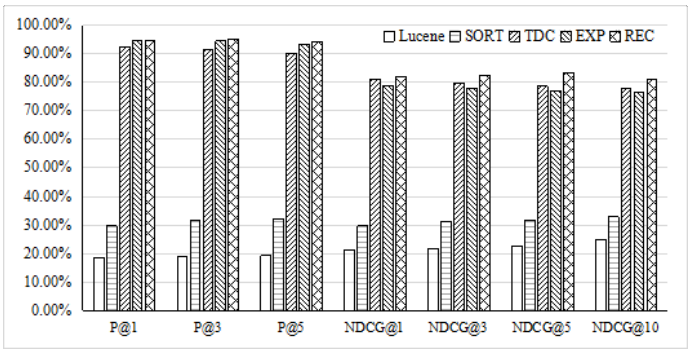

Fig. 6. Effectiveness of the five methods

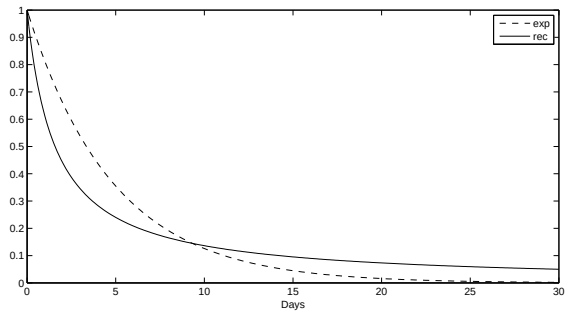

Fig. 7. The decay curve of EXP and REC

are 0.8201, 0.8256, 0.8328 and 0.8098 respectively. The performance of TDC and EXP are similar.

TABLE II
PERFORMANCE OF TDC, EXP AND REC

Measurement	TDC	EXP	REC
P@1	92.59%	94.44%	**94.44%**
P@3	91.36%	94.44%	**95.06%**
P@5	90.00%	93.33%	**94.07%**
NDCG@1	0.8122	0.7884	**0.8201**
NDCG@3	0.7976	0.7774	**0.8256**
NDCG@5	0.7856	0.7717	**0.8328**
NDCG@10	0.7788	0.7658	**0.8098**

Table II shows the performances of TDC, EXP and REC. In comparison with EXP, REC improves precision by 0.43% and enhances NDCG by 0.0463. Comparing with TDC, REC improves precision by 3.21% and enhances NDCG by 0.0285.

Figure 7 shows the decay curve of the two methods. From this figure, we can find that although the 30 days decay ratio of EXP is much smaller than REC, but the decay speed is slower than REC before 9.31 days. (*The reason for this phenomenon is that REC is a decay function which goes down faster in recent days, and slower in old days. Thus, the decay curve is rapid in the recent days.*) Thus REC can discriminate "Relevant and Useful" results from "Relevant and Newest" results in a better way, and its performance is better than EXP. The decay curve of REC is similar with the memorize curve proposed by H.Ebbinghaus [29]. It means REC can simulate the timeliness requirements of users in a better way.

The TDC method utilizes "Term Distribution Change" to determine the timeliness requirements of different queries. But in our experiment, users are willing to track information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2736064, IEEE Access

All the queries are supposed to find the newest and relevant results. Thus "Term Distribution Change" fails to predict the timeliness requirements. We estimate the Pearson correlation coefficients as same as [18], the coefficient value is -0.144. While in [18], it is -0.413. This result shows that TDC cannot predict the temporal needs in our situation. Besides, TDC is based on exponential decay function, it has the same limitation as EXP does.

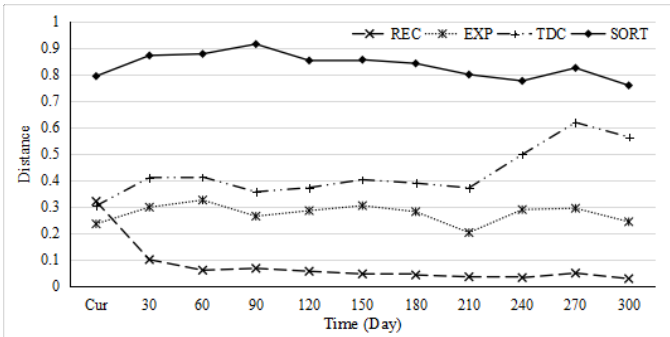

Fig. 8. The distances between the four methods and Lucene

3) Ranking Results over Time: Figure 8 shows the spearman footrule distances between Lucene and the other four baselines during different periods [30]. Spearman footrule distance is used to measure the ranking similarity between two ranking results. In this figure, the X-axis is the time period. Y-axis is the distance value. For example, the second point of the solid curve means: for ranking documents published between 30 days to 33 days, the distance between SORT and Lucene is 0.872. From this figure, we can find the distances of EXP-Lucene, SORT-Lucene and TDC-Lucene remain far over time. This phenomenon means that the temporal bias of the three methods remains stable when ranking old documents. The REC-Lucene distance becomes smaller and smaller. It means REC model takes more account of the document contents than the publish time of the documents when ranking old documents.

Table III shows three examples of the different ranking results between REC and EXP. These three queries happened in Feb. 16th 2017. We limit the publish time of the results to show the different temporal bias of the two methods. From this table, we can conclude that since EXP method always prefers newer documents, the ranking result is worse than that of REC. The top result of EXP is either out of date or less relevant. These results are useless in 2017. The REC method prefers results that are relevant to the query for old documents. Thus the top result of REC is more useful than that of EXP.

4) Web Crawler Efficiency: Our web crawler visits 1300 web sites everyday, downloads newly published pages from them. More than 300,000 new web pages are downloaded each day. And currently there are more than 100 million web pages in our index.

Figure 9 shows the efficiency of our web crawler. In this figure, X-axis is the time taken by our web crawler to discover

[2]a Chinese Olympic and world-record-holding competitive swimmer

[3]a Chinese actress, television producer and pop singer

TABLE III
THE DIFFERENT RANKING RESULT OF REC AND EXP

Query	METH	Temporal Limit	Top Result
Sun Yang[4]	REC	Aug 20-24, 2016	The whole store of the analeptic event
	EXP	Aug 20-24, 2016	Sun Yang is back to China
Fan Bingbing[5]	REC	Sep 24-27, 2016	Fan Bingbing wins the Movie Queen
	EXP	Sep 24-27, 2016	Microsoft Wheatgrass (mentioned Fan)
Kobe	REC	Nov 2-5, 2016	Kobe is coming to ShengYang
	EXP	Nov 2-5, 2016	Russell Westbrook, Kobe is mentioned

the page. From this figure, 81.87% web pages are downloaded in less than two hours, and 67.88% of web pages are downloaded in less than an hour. This means most useful web pages are discovered in less than an hour. Only 0.77% web pages costs more than sixteen hours. These web pages exist deeply in the site structure, and are less important. The results tagged as "Old" and "Useful" are likely to be downloaded slower than others. Because "Old" and "Useful" information are less important and are less likely to be appeared in home pages.

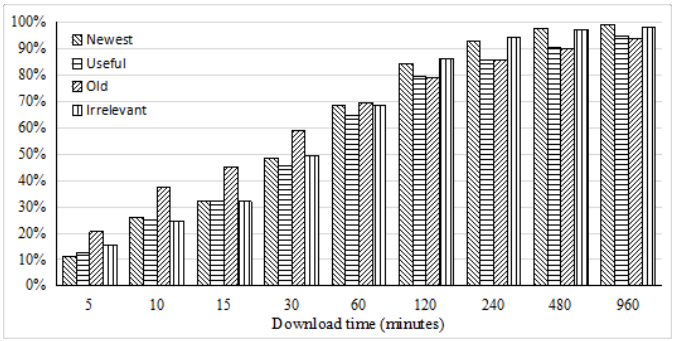

Fig. 9. The efficiency of the web crawler

V. CONCLUSION AND FUTURE WORK

In this paper, we have presented a novel type of web service called Tracking Engine. Tracking Engine is designed for users for getting a steady fix on the information they are interested or concerned. A typical usage is for mobile users to get timely information every day. The Tracking Engine mainly consists of three parts: Web Crawler, Tracking Model and Ranking Model. We have described our solution of these components. The ranking model is similar to time-based searching [7]. We have proposed a new decay function for time-based searching. Our experiment results show that our ranking model is better than previous works in terms of precision and NDCG.

Our work for the future will be the development of techniques for automatic determination of the parameter in REC. We will also doing a research on determination of the relevance of a document to the user strategy.

REFERENCES

[1] J. Ho Cheong and M.-C. Park, "Mobile internet acceptance in korea," *Internet research*, vol. 15, no. 2, pp. 125–140, 2005.

[2] L. Dong and H. Wu, "Mobile internet and regional development in china," *Environment and Planning A*, vol. 49, no. 4, pp. 725–727, 2017.

[3] A. Arasu, J. Cho, H. Garcia-Molina, A. Paepcke, and S. Raghavan, "Searching the web," *ACM Transactions on Internet Technology (TOIT)*, vol. 1, no. 1, pp. 2–43, 2001.

[4] J. Cho and H. Garcia-Molina, "The evolution of the web and implications for an incremental crawler," Stanford, Tech. Rep., 1999.

[5] N. Pawar, K. Rajeswari, and A. Joshi, "Implementation of an efficient web crawler to search medicinal plants and relevant diseases," in *Computing Communication Control and automation (ICCUBEA), 2016 International Conference on*. IEEE, 2016, pp. 1–4.

[6] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: bringing order to the web." 1999.

[7] X. Li and W. B. Croft, "Time-based language models," in *Proceedings of the twelfth international conference on Information and knowledge management*. ACM, 2003, pp. 469–475.

[8] K. Berberich, M. Vazirgiannis, and G. Weikum, "Time-aware authority ranking," *Internet Mathematics*, vol. 2, no. 3, pp. 301–332, 2005.

[9] J. Cho, S. Roy, and R. E. Adams, "Page quality: In search of an unbiased web ranking," in *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*. ACM, 2005, pp. 551–562.

[10] N. Dai and B. D. Davison, "Freshness matters: in flowers, food, and web authority," in *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2010, pp. 114–121.

[11] R. Zhang, Y. Chang, Z. Zheng, D. Metzler, and J.-y. Nie, "Search result re-ranking by feedback control adjustment for time-sensitive query," in *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*. Association for Computational Linguistics, 2009, pp. 165–168.

[12] A. Dong, Y. Chang, Z. Zheng, G. Mishne, J. Bai, R. Zhang, K. Buchner, C. Liao, and F. Diaz, "Towards recency ranking in web search," in *Proceedings of the third ACM international conference on Web search and data mining*. ACM, 2010, pp. 11–20.

[13] Y. Inagaki, N. Sadagopan, G. Dupret, A. Dong, C. Liao, Y. Chang, and Z. Zheng, "Session based click features for recency ranking." in *AAAI*, vol. 10, 2010, pp. 1334–1339.

[14] A. Dong, R. Zhang, P. Kolari, J. Bai, F. Diaz, Y. Chang, Z. Zheng, and H. Zha, "Time is of the essence: improving recency ranking using twitter data," in *Proceedings of the 19th international conference on World wide web*. ACM, 2010, pp. 331–340.

[15] N. Dai, M. Shokouhi, and B. D. Davison, "Learning to rank for freshness and relevance," in *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 2011, pp. 95–104.

[16] A. Styskin, F. Romanenko, F. Vorobyev, and P. Serdyukov, "Recency ranking by diversification of result set," in *Proceedings of the 20th ACM international conference on Information and knowledge management*. ACM, 2011, pp. 1949–1952.

[17] M. Efron and G. Golovchinsky, "Estimation methods for ranking recent information," in *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 2011, pp. 495–504.

[18] S. Cheng, A. Arvanitis, and V. Hristidis, "How fresh do you want your search results?" in *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2013, pp. 1271–1280.

[19] C. Luo, X. Li, Y. Liu, T. Sakai, F. Zhang, M. Zhang, and S. Ma, "Investigating users' time perception during web search," in *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*. ACM, 2017, pp. 127–136.

[20] K. Vierordt, *Der zeitsinn nach versuchen*. H. Laupp, 1868.

[21] R. Campos, G. Dias, A. M. Jorge, and A. Jatowt, "Survey of temporal information retrieval and related applications," *ACM Computing Surveys (CSUR)*, vol. 47, no. 2, p. 15, 2015.

[22] N. Kanhabua, R. Blanco, K. Nørvåg *et al.*, "Temporal information retrieval," *Foundations and Trends® in Information Retrieval*, vol. 9, no. 2, pp. 91–208, 2015.

[23] J. Bian, X. Li, F. Li, Z. Zheng, and H. Zha, "Ranking specialization for web search: a divide-and-conquer approach by using topical ranksvm," in *Proceedings of the 19th international conference on World wide web*. ACM, 2010, pp. 131–140.

[24] R. Baeza-Yates, F. Saint-Jean, and C. Castillo, "Web structure, dynamics and page quality," in *International Symposium on String Processing and Information Retrieval*. Springer, 2002, pp. 117–130.

[25] E. Hatcher and O. Gospodnetic, "Lucene in action," 2004.

[26] G. Salton and M. J. McGill, "Introduction to modern information retrieval," 1986.

[27] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[28] K. Järvelin and J. Kekäläinen, "Cumulated gain-based evaluation of ir techniques," *ACM Transactions on Information Systems (TOIS)*, vol. 20, no. 4, pp. 422–446, 2002.

[29] H. Ebbinghaus, *Memory: A contribution to experimental psychology*. University Microfilms, 1913, no. 3.

[30] R. Fagin, R. Kumar, and D. Sivakumar, "Comparing top k lists," *SIAM Journal on Discrete Mathematics*, vol. 17, no. 1, pp. 134–160, 2003.