Evaluating the Effectiveness of Personalized Web Search

Zhicheng Dou, Ruihua Song, Ji-Rong Wen, and Xiaojie Yuan

Abstract—Although personalized search has been under way for many years and many personalization algorithms have been investigated, it is still unclear whether personalization is consistently effective on different queries for different users and under different search contexts. In this paper, we study this problem and provide some findings. We present a large-scale evaluation framework for personalized search based on query logs and then evaluate five personalized search algorithms (including two click-based ones and three topical-interest-based ones) using 12-day query logs of Windows Live Search. By analyzing the results, we reveal that personalized Web search does not work equally well under various situations. It represents a significant improvement over generic Web search for some queries, while it has little effect and even harms query performance under some situations. We propose click entropy as a simple measurement on whether a query should be personalized. We further propose several features to automatically predict when a query will benefit from a specific personalization algorithm. Experimental results show that using a personalization algorithm for queries selected by our prediction model is better than using it simply for all queries.

Index Terms—Web search, personalization, information filtering, performance evaluation.

1 Introduction

One criticism of search engines is that when queries are issued, most return the same results to users. In fact, the vast majority of queries to search engines are short [1], [2] and ambiguous [3], [4]. Different users may have completely different information needs and goals when using precisely the same query [2], [5], [6], [7]. For example, a biologist may query "mouse" to get information about rodents, while programmers may use the same query to find information about computer peripherals. When such a query is issued, search engines will return a list of documents that mix different topics, as shown in Table 1. It takes time for a user to choose which information he/she wants. On another query of "free mp3 download," although most users find websites to download free mp3s, their selections can diverge: one may choose the website www.yourmp3.net, while another may prefer the website www.seekasong.com.

Personalized search is considered a solution to address these problems, since it can provide different search results based upon the preferences of users. Various personalization strategies, which include [5], [8], [9], [10], [11], [12], [13], [14], and [15], have been proposed. However, they are far from optimal. One problem of current personalized search is that most proposed algorithms are uniformly applied to all users and queries. We argue that queries should *not* be handled in the same general manner:

First, personalization may lack effectiveness on some queries, and thus, there is no need of it for these queries; this has also been found by Teevan et al. [7]. For example, on the query "mouse" mentioned above, using personalization based on topical interests of users (for example, the one proposed by Chirita et al. [16]), we could achieve greater relevance for individual users than a common Web search. Beyond all doubt, the personalization is of great benefit to users in this case. Contrarily, for the query "Google," which is a typical navigational query as defined by Broder [17] and Lee et al. [18], almost all users consistently select a link to Google's homepage. Therefore, none of the personalized strategies could provide obvious benefits to users.

Second, personalization algorithms have strengths and weaknesses for different queries. For example, topical-interest-based personalization, which leads to better performance for the query "mouse," is ineffective for the query "free mp3 download." Actually, relevant documents for query "free mp3 download" are mostly classified into the same topic categories, and topical-interest-based personalization has no way to filter out desired documents. In such a case, simply leveraging pages visited by this user in the history may achieve better performance.

Third, the effectiveness of personalization algorithms may vary due to various search contexts. For example, it might prove difficult to learn the interests of users who have done few searches. Even if search histories are enough to infer general user interests, users often search for short-term information needs that may be inconsistent with general user interests, as Shen et al. [19] found. In such cases, long-term user profiles are useless or even harmful, whereas a short-term query context is more useful.

Another problem is that many personalization algorithms are proposed by considering only positive aspects and are evaluated upon a small number of manually selected queries. Little investigation has been done on how personalization strategies perform under real-world search engine conditions faced by users. In this paper, we address these

- *Z. Dou, R. Song, and J.-R. Wen are with the Microsoft Research Asia, 49 Zhichun Road, Haidian District, Beijing 100190, P.R. China. E-mail: {zhichdou, rsong, jrwen}@microsoft.com.*
- *X. Yuan is with the Nankai University, 94 Weijin Road, Nankai District, Tianjin 300071, P.R. China. E-mail: yuanxj@nankai.edu.cn.*

Manuscript received 24 July 2007; revised 22 Apr. 2008; accepted 31 July 2008; published online 15 Aug. 2008.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-2007-07-0381.
Digital Object Identifier no. 10.1109/TKDE.2008.172.

1041-4347/09/$25.00 © 2009 IEEE Published by the IEEE Computer Society

TABLE 1
Top 10 Search Results for Query "Mouse" in Windows Live Search (www.live.com) in July 2007

ID	Title; URL	Topic
1	Apple - Mighty Mouse; http://www.apple.com/mightymouse/	Computer peripheral
2	Mouse - Wikipedia, the free encyclopedia; http://en.wikipedia.org/wiki/Mouse	Rodent
3	Mouse (computing) - Wikipedia, the free encyclopedia; http://en.wikipedia.org/wiki/Computer_mouse	Computer peripheral
4	mouse.org - Home; http://www.mouse.org/	Computer peripheral
5	mouse.org - About; http://www.mouse.org/about	Computer peripheral
6	Hardware Image Gallery: Microsoft Mouse Products; http://www.microsoft.com/presspass/gallery/hardware-mouse.mspx	Computer peripheral
7	Microsoft Hardware Mouse and Keyboard -Home Page ; http://www.microsoft.com/hardware/mouseandkeyboard/default.mspx	Computer peripheral
8	YouTube - Giant centipede eating mouse; http://www.youtube.com/watch?v=8CL2hetqpfg	Rodent
9	MGI_3.51 - Mouse Genome Informatics; http://www.informatics.jax.org/	Rodent
10	Mickey Mouse | Mouse Stuff; http://disney.go.com/characters/mickey/html/stuff/index.html	Cartoon animal Mickey Mouse

problems and make some contributions. We develop a large-scale personalized search evaluation framework based on real-world query logs. In this framework, different personalized reranking algorithms are simulated, and search accuracy is evaluated by real user clicks. The framework enables us to evaluate personalization approaches on a large-scale data set. We implement two click-based personalized search methods and three topical-interest-based methods, evaluate the five approaches in the proposed framework using 12 days of query logs from the Windows Live search engine [20], and provide detailed analysis on experimental results.

We reveal that personalized Web search has different levels of effectiveness for different queries, users, and search contexts. We propose click entropy to measure whether users have diverse information needs by issuing a query. Experimental results show that personalization brings significant search accuracy improvements on queries with a large click entropy and has little effect on queries with a small click entropy. Personalization algorithms can even harm search accuracy on some queries. Therefore, we conclude that queries should not be personalized identically.

Since a specific personalization algorithm cannot improve ranking accuracy for all queries and it even harms search accuracy under some situations, we propose several features to automatically predict whether an algorithm should be used to personalize a given query. We experiment with predicting when Web search results can be improved using a personalization method based on a user's long-term interests. Experimental results show that using the personalization algorithm for queries selected by our prediction model is better than using it simply for all queries.

The remaining sections are organized as follows: In Section 2, we discuss related work. We present an evaluation framework of personalization algorithms in Section 3. In Section 4, we briefly describe and evaluate several personalization algorithms. In Section 5, we provide detailed statistics of the data set used in our experiments. In Section 6, we compare and analyze experimental results. In Section 7, we propose click entropy and other features to predict whether a personalization algorithm should be used for given queries and experiment with predicting when Web search results can be improved using a personalization method based on a user's long-term interests. We then conclude our work in Section 8.

2 Related Work

There are three categories of work related to this paper: personalizing Web search algorithms, analysis of personalized Web search, and query performance prediction. We will introduce them in the following sections separately.

2.1 Personalized Web Search Algorithms

There have been several prior attempts on personalizing Web search. A comprehensive survey on personalized search can be found in [21]. In the following sections, we will summarize previous personalized search strategies, including personalized search based on content analysis, personalized search based on the hyperlink structure of the Web, and personalized search based on user groups.

2.1.1 Personalized Search Based on Content Analysis

One approach of personalized search is to filter or rerank search results by checking content similarity between returned web pages and user profiles. User profiles store approximations of user interests. User profiles are either specified by users themselves [9], [16] or are automatically learnt from a user's historical activities. As the vast majority of users are reluctant to provide any explicit feedback on search results and their interests [22], many works on personalized Web search focus on how to automatically learn user preferences without the user being required to directly participate [5], [9], [15], [23]. In terms of how user profiles are built, there are two groups of works: *topical categories* [9], [15], [24] or *keyword lists* (bags of words) [5], [10], [13], [23], [25].

Several approaches represent user interests by using topical categories. In [9], [16], [26], [27], [28], and [29], a user profile is usually structured as a concept/topic hierarchy. User-issued queries and user-selected snippets/documents are categorized into concept hierarchies that are accumulated to generate a user profile. When the user issues a query, each of the returned snippets/documents is also classified. The documents are reranked based upon how well the document categories match user interest profiles.

Some other personalized search approaches use lists of keywords to represent user interests. Sugiyama et al. [23] built user preferences as vectors of distinct terms and constructed them by aggregating past preferences, including both long-term and short-term preferences. Shen et al. [5] first used language modeling to mine contextual information from a short-term search history. Tan et al. [10] then used the method to mine context from a long-term search history. Teevan et al. [13] and Chirita et al. [25] exploit rich models of user interests, built from both search-related information and other information about the user, including documents and e-mails that the user has read and created. In the work of Liu et al. [15], [24], keywords are associated with categories, and thus, user profiles are represented by a hierarchical category tree based on keyword categories.

In this paper, we implement several topical-interest-based personalization strategies, similar to those in [9], [16], and [29], and user profiles are also automatically learned from users' past queries and click-throughs in search engine logs.

2.1.2 Personalized Search Based on the Hyperlink Structure of the Web

Many personalized Web search strategies based on the hyperlink structure of the Web have also been investigated. Personalized PageRank (PPR), which is a modification of the global PageRank algorithm, was first proposed for personalized Web search by Page et al. [30]. Haveliwala [31] used multiple PPR scores, one for each main topic of ODP, to enable "topic-sensitive" Web search. Jeh and Widom [11] gave an approach that could scale well with the size of hub vectors to realize personalized search based on Topic-Sensitive PageRank. Sarlós et al. [32] used lossy representation of large vectors by either rounding or sketching and made PPR usable in real-world applications. Tanudjaja and Mui [33] extended the well-known HITS algorithm by artificially increasing the authority and hub scores of pages marked relevant by the user in previous searches. Most recently, Lee et al. [18] developed a method to automatically estimate hidden user interests based on Topic-Sensitive PageRank scores of the user's past clicked pages. In this paper, we do not implement these kinds of methods because we have no necessary Web collection.

2.1.3 Personalized Search Based on User Group

In most of the above personalized search strategies, only the information provided by a user himself/herself is exploited to create user profiles. These are also some strategies that incorporate the preferences of a group of users to accomplish personalized search. In these approaches, search histories of users who have similar interests with a test user are used to refine the search. Collaborative filtering (CF) is a typical group-based personalization method and has been used in personalized search by Sugiyama et al. [23] and Sun et al. [14]. Sugiyama et al. [23] constructed user profiles based on a modified CF algorithm [34]. Sun et al. [14] proposed a novel method, named CubeSVD, to apply personalized Web search by analyzing correlations among users, queries, and web pages in click-through data. In this paper, we also introduce a method that incorporates click histories of a group of users with similar topical affinities to personalize Web search.

2.2 Analysis of Personalized Web Search

In this paper, we reveal that personalization should not be used for all queries in the same manner. Some researchers have also noticed that personalization varies in effectiveness for different queries. For instance, Teevan et al. [7] suggested that not all queries would be handled in the same manner. For less ambiguous queries, current Web search ranking might be sufficient, and thus, personalization is unnecessary. Chirita et al. [16], [25], [35] divided test queries into three types: clear queries, semiambiguous queries, and ambiguous queries. They concluded that personalization significantly increased output quality for ambiguous and semiambiguous queries, but for clear queries, one would prefer a common Web search. Tan et al. [10] divided queries into fresh queries and recurring queries. They found that the recent history tended to be much more useful than the remote history, especially for fresh queries, whereas the entire history was helpful for improving the search accuracy of recurring queries. These conclusions inspired our work of detailed analysis on these kinds of problems.

Teevan et al.'s recent work [36] is quite relevant to the work in this paper. They also revealed that personalization does not work equally well on all queries. They examined the variability in user intent using both implicit and explicit measures and further proposed several features to predict variation in user intent. In Section 7 of this paper, we make further investigations on this direction. We build predictive models to directly identify the queries that will benefit from a specific personalization algorithm.

2.3 Query Performance Prediction

There is much existing work on predicting query performance [4], [37], [38], [39], [40]. The characteristics of query and/or search results have been used to predict the performance of a generic search on a query. For example, Zhou and Croft [40] developed measures for both content-based tasks and Named-Page finding tasks. Different from previous work, we focus on predicting the query performance of personalized Web search. From this point of view, Teevan et al.'s work [36] is quite relevant because it also predicts the query performance of personalized Web search.

3 Experiment Methodology

A typical evaluation method used in existing personalized search research is to conduct user studies [5], [6], [7], [10], [15], [16], [23], [25], [29]. Usually, a certain number of people participate in evaluating a personalized search system over several days. User profiles are manually specified by participants themselves [16] or automatically learned from search histories. To evaluate the performance of personalized search, each participant is required to issue a certain number of test queries and determine whether each result is relevant. An advantage of this approach is that the relevance of documents can be explicitly judged by the participants. Unfortunately, there are some drawbacks in this method. Constraints on the number of participants and test queries may bias evaluation results on accuracy and reliability of the personalization algorithm.

We propose a framework that enables large-scale evaluation of personalized search. In this framework, we use click-through data that is recorded in search engine logs to simulate user experiences in Web search. In general, when a user issues a query, the user usually checks documents in a result list from top to bottom [41], [42]. The user clicks one or more documents that look relevant and skips those documents that the user is not interested in. If a specific personalization method can rerank relevant documents for a user higher in results list, the user would be more satisfied. Therefore, we utilize user clicks as relevance judgments to evaluate search accuracy. Since click-through data can be collected at low cost, it is possible to do large-scale evaluation under this framework. Furthermore, click-through data reflects the real-world distribution of queries, users, and search scenarios. Thus, using click-through data is closer to real cases in evaluating personalized search than user surveys.

One main concern about this evaluation framework is the position bias (i.e., presentation bias) [41], [43] in click-through data. User clicks are highly biased toward documents that are ranked at the top of the rank list. Since we just use clicks made on the original rank list to evaluate the quality of the personalized rank list, this may include bias: 1) some relevant documents might be ranked lower in the original result list due to problems of the original ranking algorithm, and 2) the original ranking algorithm orders the results across all content topics based on their overall relevance. Documents that are very relevant in a specific topic may be ranked lower in the original result list. In these two cases, documents that are relevant to a user may be ranked lowly in the original rank list and receive no clicks. Evaluation based on user clicks in such cases is inaccurate. We will adapt our framework to enable online evaluation to avoid the problem of presentation bias in future work.

Another concern is the snippet-content mismatch problem: a user's click decision is usually not motivated by the document content itself but the result snippet. User clicks may fail to reflect the real relevance of a document when the snippet does not fully reflect the document content. A document that is highly relevant but displays a bad snippet may receive few clicks, and a document that is not relevant but shows an appealing snippet may receive many clicks. A possible solution is to take into account user's browsing behavior when viewing a page, such as dwelling time, page scrolling, mouse click, and mouse movement. We will investigate this in future work.

Despite the above bias and noise, click-through data contains much useful information about relevance and user preferences. Our framework is still useful to evaluate the approximate top precision of personalized Web search strategies, especially when experimenting with a large number of queries. Currently, it is the best method we can use to enable large-scale evaluation of personalized Web search.

In the evaluation framework, we use query logs of Windows Live Search to simulate and evaluate personalized reranking strategies. We organize a log entry for a query as the format shown in Fig. 1. In Windows Live Search query logs, each *user* is identified by "Cookie GUID" that remains the same in a machine as long as a cookie is not cleaned. For each *query*, the Windows Live search engine logs the query string and all click-through information, including clicked web pages and their corresponding ranks. A "Browser GUID" is assigned when a browser is opened and expired when the browser is closed. "Browser GUID" is used as a simple identifier of a *session* that contains a series of related queries made by a user within a small range of time. A session is usually meaningful in capturing a user's attempts to fulfill certain information needs [1], [2].

Time, Cookie GUID, Query String, Browser GUID
{
  (Position$_1$,URL$_1$), (Position$_2$,URL$_2$), ... (Position$_n$,URL$_n$)
}

Fig. 1. Representation of a log entry.

3.1 Reranking Evaluation Framework

In the proposed framework, we first download search results from the Windows Live search engine. Then, we use a selected personalization algorithm to rerank search results. Finally, clicked URLs for queries in a test set are used as ground truth in evaluating reranking performance. To be specific, given a personalization algorithm and a log entry in the test set, we propose to use the following steps to simulate personalized reranking and to conduct evaluation:

First, download the top 50 search results from the Windows Live search engine for the query string. We denote downloaded result items with U and denote the ranking list with $\tau_{Original}$. We download only the top 50 search results because we find most users never look beyond the top 50 entries in search logs.

Second, compute a personalized score for each item $x_i \in U$ using a given personalization algorithm and then generate a new rank list $\tau_{Personalized}$ with respect to U. Result items in $\tau_{Personalized}$ are sorted by personalized scores in descending order.

Third, combine the two rank lists $\tau_{Original}$ and $\tau_{Personalized}$ by the well-known Borda's ranking aggregation method [44], [45] to generate a final rank list τ_{Rerank}. τ_{Rerank} will be returned to users in personalized search. We incorporate original document ranks into the final ranking to enhance the stability of personalization algorithms for the following reasons. First, some personalization algorithms may generate identical personalized scores for many results. For example, for click-based personalization algorithms that will be introduced in Section 4, all documents that are not clicked in the past will get the same personalized score 0. Original ranking will help decide ranks of these documents. Second, some personalization algorithms may ignore the original content quality of the results. For example, the topical-interest-based methods that will be introduced in Section 4 only consider the topic importance of the results. Original ranking will help make sure reasonable content quality of the final result list. Please note that we do not use a score-based aggregation method but a rank-based one because we have no way to obtain relevance scores from the Windows Live search engine.

Fourth, get the ranks of clicked URLs in a log entry and use the measurements introduced in Section 3.2 to evaluate the performance of τ_{Rerank}.

3.2 Evaluation Measurements

We use the *rank scoring* metric used by Sun et al. [14] and Breese et al. [34] to evaluate the accuracy of personalized search. The rank scoring metric is used to evaluate the effectiveness of CF systems that return an ordered list of recommended items [34]. Sun et al. [14] use it to evaluate the retrieval performance of personalized Web search.

The expected utility of a ranked list of documents is defined as

$$R_s = \sum_j \frac{\delta(s,j)}{2^{(j-1)/(\alpha-1)}},$$

where j is the rank of a document, $\delta(s,j)$ is 1 if the $j\text{th}$ document is clicked for the test query s and 0 if not clicked. α is a parameter set as five as the authors suggest. The final rank scoring reflects the utility of all test queries:

$$R = 100 \frac{\sum_s R_s}{\sum_s R_s^{Max}}. \qquad (1)$$

Here, R_s^{Max} is the maximum utility when all clicked documents move to the top of a ranked list. A larger rank scoring value means better retrieval performance.

4 Personalization Algorithms

We implement several personalization algorithms that we introduced in Section 2 and propose straightforward algorithms of using historical clicks in personalization. In general, these algorithms are used to rerank search results by computing a personalized score $S(q,p,u)$ for each document p in the results returned to user u for query q. We organize these strategies as person-level and group-level ones in this section.

4.1 Person-Level Reranking

4.1.1 Historical Click-Based Algorithm

We suppose that for a query q submitted by a user u, the web pages frequently clicked by u in the past are more relevant to u than those seldom clicked by u. Thus, a personalized score on page p can be computed by

$$S^{P-Click}(q,p,u) = \frac{|Clicks(q,p,u)|}{|Clicks(q,\bullet,u)| + \beta}. \qquad (2)$$

Here, $|Clicks(q,p,u)|$ is the number of clicks on web page p by user u for query q in the past. $|Clicks(q,\bullet,u)|$ is the total number of clicks for query q by u, and β is a smoothing factor ($\beta = 0.5$ in this paper). Actually, $|Clicks(q,\bullet,u)|$ and β are used to normalize $|Clicks(q,p,u)|$.

A disadvantage of this approach is that it is not applicable for new queries that the user has never asked. We find that in our data set, about 1/3 of the test queries are issued for more than one time by the same user. This approach would only benefit these queries. Another disadvantage of P-Click is that it may impede the discovery of newly available results because old clicked documents will be ranked to the top of result list. A feasible solution to this problem is providing personalized results in a separated list (for example, provide a short list in the right side bar of the results page) and preserve the original ranking. Another solution is to randomly push newly available results toward the top of the list every once in a while or to combine them with previously clicked documents.

We denote this approach with **P-Click**.

4.1.2 User-Topical-Interest-Based Algorithms

As described in Section 2, much previous work employs user interest to personalize search results [9], [15], [16]. In this paper, we also implemented a personalization method based on long-term user topical interests (we denote this method with **L-Topic**). The user's interests $\boldsymbol{c}_l(u)$ are represented as a vector of 67 predefined topic categories modified from the second-level categories provided by KDD Cup-2005 [46]. When a user submits a query, each returned web page is first mapped to a category vector. Then, the similarity between the user profile vector and the page category vector is computed:

$$sim(\boldsymbol{c}_l(u), \boldsymbol{c}(p)) = \frac{\boldsymbol{c}_l(u) \cdot \boldsymbol{c}(p)}{\|\boldsymbol{c}_l(u)\| \|\boldsymbol{c}(p)\|}.$$

Here, $\boldsymbol{c}(p)$ is the category vector of web page p. $\boldsymbol{c}(p)$ is generated by a similar text classifier used by Shen et al. [47]. Given a web page p, the classifier returns the top six categories that p most likely belongs to with corresponding confidences. Each element $c(p)_i$ of $\boldsymbol{c}(p)$ is the confidence returned by the classifier. If category i is not among the six returned categories, we set $c(p)_i = 0$.

User profile $\boldsymbol{c}_l(u)$ is computed based upon his/her past clicked web pages by the following equation:

$$\boldsymbol{c}_l(u) = \sum_{p \in \mathcal{P}(u)} P(p|u) w(p) \boldsymbol{c}(p).$$

Here, $\mathcal{P}(u)$ is the collection of web pages that were visited by user u in the past. $P(p|u)$ can be thought of as the probability that user u clicks web page p, i.e.,

$$P(p|u) = \frac{|Clicks(\bullet,p,u)|}{|Clicks(\bullet,\bullet,u)|}.$$

Here, $|Clicks(\bullet,\bullet,u)|$ is the total number of times that u clicked, and $|Clicks(\bullet,p,u)|$ is the number of times that u clicked on web page p. $w(p)$ is an impact weight for page p when we generate user profiles. We assume that the web pages that have been clicked by many users are less important in building distinguishable user profiles. Thus,

$$w(p) = \log \frac{|\mathcal{U}|}{|\mathcal{U}(p)|}.$$

$|\mathcal{U}|$ is the number of total users; $|\mathcal{U}(p)|$ is the number of users who have ever visited web page p.

The similarity between user interests and a web page is used to rerank search results. To reduce the instability of personalization, only the web pages that are similar enough with user interests are reranked. We use a threshold t to control whether a web page should be reranked. The personalization score of document p is defined as

$$S^{L-Topic}(q,p,u) = \begin{cases} sim(\boldsymbol{c}_l(u), \boldsymbol{c}(p)) & \text{if } sim(\boldsymbol{c}_l(u), \boldsymbol{c}(p)) \geq t \\ 0 & \text{if } sim(\boldsymbol{c}_l(u), \boldsymbol{c}(p)) < t \end{cases} \quad t \in [0,1]. \qquad (3)$$

Contrasted to a long-term user profile, [5] investigated a short-term user profile and found that it is more useful for improving search in an ongoing session. We use clicks on previous queries in an ongoing session to build a short-term user profile and then exploit short-term interests to personalize search. Such an approach is denoted with **S-Topic**. A short-term user profile $\boldsymbol{c}_s(u)$ is computed as

$$\boldsymbol{c}_s(u) = \frac{1}{|\mathcal{P}_s(q)|} \sum_{p \in \mathcal{P}_s(q)} \boldsymbol{c}(p).$$

$\mathcal{P}_s(q)$ is the collection of visited pages on previous queries in this session. The similarity between short-term user interests and a web page is defined as

$$sim(\boldsymbol{c}_s(u), \boldsymbol{c}(p)) = \frac{\boldsymbol{c}_s(u) \cdot \boldsymbol{c}(p)}{\|\boldsymbol{c}_s(u)\| \|\boldsymbol{c}(p)\|}.$$

Similar to (4), a personalized score of page p by using a short-term profile is computed as

$$S^{S-Topic}(q,p,u) = \begin{cases} sim(\boldsymbol{c}_s(u), \boldsymbol{c}(p)) & \text{if } sim(\boldsymbol{c}_s(u), \boldsymbol{c}(p)) \geq t \\ 0 & \text{if } sim(\boldsymbol{c}_s(u), \boldsymbol{c}(p)) < t \end{cases} t \in [0,1]. \quad (4)$$

We can also fuse the long-term personalized score and the short-term personalized score by a simple linear combination:

$$sim(\boldsymbol{c}_{ls}(u), \boldsymbol{c}(p)) = \theta sim(\boldsymbol{c}_s(u), \boldsymbol{c}(p)) + (1-\theta) sim(\boldsymbol{c}_l(u), \boldsymbol{c}(p)).$$

Thus,

$$S^{LS-Topic}(q,p,u) = \begin{cases} sim(\boldsymbol{c}_{ls}(u), \boldsymbol{c}(p)) & \text{if } sim(\boldsymbol{c}_{ls}(u), \boldsymbol{c}(p)) \geq t \\ 0 & \text{if } sim(\boldsymbol{c}_{ls}(u), \boldsymbol{c}(p)) < t \end{cases} t \in [0,1]. \quad (5)$$

We denote this hybrid approach with **LS-Topic**. Methods L-Topic, S-Topic, and LS-Topic are generally called *topical-interest-based* methods for short in this paper.

4.2 Group-Level Reranking

We implement a K-Nearest Neighbor CF algorithm as a representative of group-based personalization. Due to sparse data, we find that applying traditional CF methods on Web search is inadequate. Instead, we compute user similarity based on long-term user profiles:

$$sim(u_1, u_2) = \frac{\boldsymbol{c}_l(u_1) \cdot \boldsymbol{c}_l(u_2)}{\|\boldsymbol{c}_l(u_1)\| \|\boldsymbol{c}_l(u_2)\|}.$$

The K-Nearest neighbors are obtained based on the user similarity:

$$\mathcal{S}_u(u_a) = \{u_s | rank(sim(u_a, u_s)) \leq K\}.$$

Then, we use the historical clicks made by all similar users in a group to rerank the search results:

$$S^{G-Click}(q,p,u) = \frac{\sum_{u_s \in \mathcal{S}_u(u)} sim(u_s, u) |Clicks(q,p,u_s)|}{\beta + \sum_{u_s \in \mathcal{S}_u(u)} |Clicks(q, \bullet, u_s)|}. \quad (6)$$

We denote this group-level approach with **G-Click**.

4.3 Complexity and Performance of Algorithms

In this section, we briefly discuss the complexity and performance of deploying the above algorithms onto real search engines. Personalized Web search can be implemented on either the server side (in the search engine) or the client side (in the user's computer or a personalization agent) [21]. For client-side personalization, user information is collected and stored on the client side. The overhead in computation and storage for personalization can be distributed among the clients. Because the size of the browsing history of one user is usually small, the cost of storage and computation for generating user profiles on each client is low. For the server-side implementation, we may need to process some data in advance. More specifically, we may need the following data processes to make sure of the high efficiency of proposed algorithms:

1. The category vector of a page is calculated offline after the page is crawled from the Web and is stored as additional metadata of the page.
2. The collection of raw click-through data is real time in search engine log systems. Grouped click-through data, e.g., $|Clicks(q,p,u)|$ and $|Clicks(q,\bullet,u)|$, can be periodically updated based on newly available raw click-through data. The length of the update period will affect the accuracy of user profiles.
3. Topical-interest-based user profile $\boldsymbol{c}_l(u)$ is periodically updated together with grouped click-through data.
4. User similarity $sim(u_1, u_2)$ is updated after $\boldsymbol{c}_l(u)$ is updated.
5. Click-through data on current user sessions are temporarily cached.

Short-term profile $\boldsymbol{c}_s(u)$ is generated online based on cached session data. Based on the above processes, the proposed algorithms can work efficiently. The storage and computation for these offline and online processes may need the help of distributed and parallel systems [48], [49].

5 Data Set

We collect a set of Windows Live query logs for 12 days in August 2006 in our experiments. As the entire log set is too huge, we randomly sample 10,000 distinct users (identified by "Cookie GUID") in the US who issued at least one query on 19 August 2006. Then, we extract all click-through data of these users through the 12 days. Please note that queries without any clicks are excluded from the data set. The entire data set is split into two subsets: one for training and one for testing. The training set contains logs of the first 11 days, while logs of the last day are used as the test set. Table 2 summarizes the statistics.

By randomly sampling users, we expect that this data set is a representative set of the whole logs. We analyze the data set and find that it has the characteristics that are similar to those in [1], [50], [51], [52], and [53]. Due to space limitations, we skip detailed statistics in this paper.

6 Performance of Proposed Algorithms

In this section, we will describe detailed evaluation experiments and results of the five personalized search strategies introduced in Section 4. As Windows Live has

TABLE 2
Basic Statistics of the Data Set

Item	ALL	Training	Test
#days	12	11	1
#users	10,000	10,000	1,792
#queries	55,937	51,334	4,639
#distinct queries	34,203	31,777	3,465
#clicks	93,566	85,642	7,924
#clicks/#queries	1.6727	1.6683	1.7081
#sessions	49,839	45,981	3,865

updated ranking results when we download search results, we fail to find clicked URLs in the search results for 676 queries out of a total 4,639 test queries. We exclude these queries in our experiments. Furthermore, we observe that users select only the top results without any jump for 57 percent (2,256/3,963) of the remaining queries. In other words, the original search method WEB has performed the best in our proposed framework. Thus, personalization cannot provide any improvements. We call these queries *optimal queries* and the other 1,707 queries *nonoptimal queries*.

In our comparison experiments, we take the performance of original Web search method without any personalization as a baseline. We denoted it with "WEB."

6.1 Overall Performance of Strategies

Table 3 summarizes the overall performance of all five personalization strategies. Please note that we set $K = 50$ for method G-Click, $t = 0.8$ for L-Topic and S-Topic, and $\theta = 0.5$ and $t = 0.8$ for LS-Topic. This table shows that methods G-Click and P-Click outperform the baseline method WEB on the whole. For instance, for the nonoptimal queries, method P-Click significantly improves $(p < 0.01)$ rank scoring by 3.69 percent over method WEB. Also, method G-Click significantly improves $(p < 0.01)$ method WEB in terms of rank scoring by 3.62 percent. Even on all test queries that include both optimal and nonoptimal queries, P-Click and G-Click methods also have significant rank scoring improvements (1.39 percent and 1.37 percent) over WEB. Therefore, we can safely conclude that click-based personalization methods can generally improve Web search performance.

Methods P-Click and G-Click have no significant difference with regard to performance. In our experiments, we sample 10,000 users and select the 50 most similar users for each test user in the G-Click approach (we also try methods to select 20 and 100 users, but they show little significant difference). By reason of high sparsity of user queries, selected similar users may have few search histories on the queries submitted by a test user. This causes group-level personalization not to perform significantly better than person-level personalization. If more day logs and more users are available, method G-Click perhaps performs better than P-Click.

TABLE 3
Overall Performance of Personalization Strategies

Method	All queries		Non-optimal queries		Optimal queries	
WEB	69.4669		47.2623		**100.0000**	
P-Click	**70.4350**	**+1.39%**	**+49.0051**	**+3.69%**	99.9029	-0.10%
L-Topic	69.0445	-0.61%	47.2570	-1.00%	99.0040	-1.00%
S-Topic	68.0799	-2.00%	46.5008	-1.61%	97.7529	-2.25%
LS-Topic	69.0578	-0.59%	47.2486	-0.03%	99.0471	-0.95%
G-Click	70.4168	+1.37%	48.9728	+3.62%	99.9040	-0.10%

For G-Click, $K = 50$*. For LS-Topic,* $\theta = 0.5$ *and* $t = 0.8$*.*

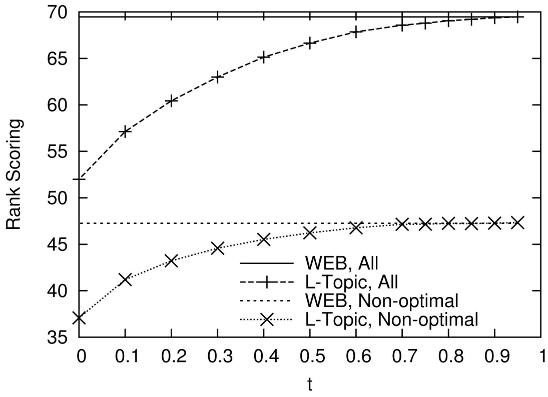


Fig. 2. Performance of algorithm L-Topic.

Table 3 also shows that topical-interest-based strategies perform less well than click-based strategies and the baseline. Topical-interest-based strategies harm search performance for many queries, especially those of optimal queries. Actually, method L-Topic lowers the search performance of optimal queries by 1.00 percent, and S-Profile lowers it by 2.25 percent. These results indicate that topical-interest-based strategies are less stable than click-based ones. Figs. 2 and 3 further plot the performance of methods L-Topic and S-Topic with similarity threshold t changing. The baseline of method WEB is also plotted in these figures. We find that methods L-Topic and S-Topic perform less well for almost all settings of t. The worst performance occurs when $t = 0$. The big drop indicates that many documents are wrongly reranked using these methods. When similarity threshold t becomes larger, which means that fewer documents are reranked, the performance increases but is still a worse method than WEB. We show the performance of method LS-Topic with different settings of t and θ in Fig. 4. The results show that method LS-Topic also lowers the performance of method WEB.

We compute rank scoring improvement for each test query and then plot the distributions for each personalization algorithm in Fig. 5. It is found that methods L-Topic, S-Topic, and LS-Topic $(\theta = 0.5, t = 0.8)$ improve search

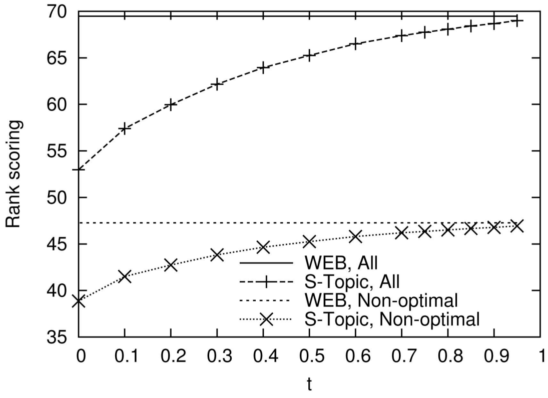


Fig. 3. Performance of algorithm S-Topic.

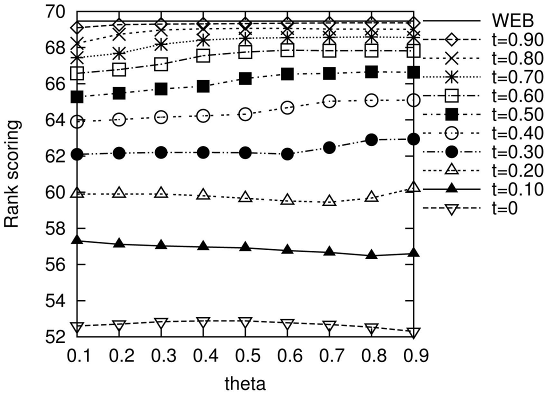


Fig. 4. Performance of algorithm LS-Topic.

performance for many queries, but they harm the performance on many more queries. That results in worse performance on the average. In contrast, click-based methods P-Click and G-Click are more stable. Only a small number of queries get worse, whereas a majority of queries are improved. This indicates that the straightforward implementation of topical-interest-based strategies we employ in this paper does not work well. Here, we try to give some possible reasons. First, as indicated by Shen et al. [5], though a user may have general long-term interests or preferences for information, often, the user is searching for documents to satisfy a short-term information need that may be inconsistent with his/her general interests. For example, a user is looking for information about a medicine for headaches just because the user feels sick that day, rather than because the user is a physician. In such cases, the user's long-term topical interests are unlikely to be helpful and could be harmful. Second, the user search history inevitably contains a lot of noisy information that is irrelevant or even harmful to the current search, as indicated by [10]. In our experiments, we simply use all historical user searches to learn user profiles without distinguishing relevant information from irrelevant data. It may cause the topical-interest-based personalization strategies to be unstable. Third, we use only 12-day search logs in our experiments. In our data set, most users have less than 20 queries in search histories. User profiles built upon such short search histories may be inconsistent with their real long-term interests. It may also cause the algorithms to generate bad results for some queries. Finally, we also do no complex normalization or smoothing in generating user profiles. As these strategies are not the optimal versions, we will complete additional investigations to try to improve performance in future work.

TABLE 4
Performance on Repeated Queries

Method	All queries		Non-optimal queries	
	Repeated	Not-rep.	Repeated	Not-rep.
WEB	84.7758	**59.9799**	46.6285	**47.4013**
P-Click	**87.3162**	59.9799	**55.9090**	47.4013
L-Topic	84.8394	59.2563	48.4746	46.9741
S-Topic	84.4529	57.9335	48.1471	46.1184
LS-Topic	84.8485	59.2722	48.3539	46.9919
G-Click	87.2685	59.9799	55.7377	47.4013

TABLE 5
Performance on Self-Repeated Queries

Method	All queries		Non-optimal queries	
	Self-repeated	Not-rep.	Self-repeated	Not-rep.
WEB	85.6337	63.2697	45.7215	47.4858
P-Click	**89.1264**	63.2697	**59.4750**	47.4858
L-Topic	85.7578	62.6378	48.4923	47.0778
S-Topic	85.4445	61.4236	47.7202	46.3240
LS-Topic	85.7508	62.6589	48.1993	47.1107
G-Click	89.0627	**63.2793**	59.1086	**47.5025**

6.2 Performance on Repeated Queries

In our data set, we observe that about 46 percent of test queries are once repeated, and 33 percent of queries are repeated by the same user. It shows that users often resubmit a query and review the results they have searched. Teevan et al. [54] have observed a similar refinding behavior. They found that such behavior is common, and thus, repeated clicks can be predicted based upon a user's historical queries and clicks. As methods P-Click and G-Click are based on historical clicks, the high repetition ratio in real query logs explains why they work well.

Table 4 shows detailed performance for the repeated queries and other queries. Table 5 shows detailed results for the self-repeated queries and other queries. We find that P-Click and G-Click have significant improvements over the WEB method for self-repeated queries. It is reasonable because a user usually has highly consistent selections with the past when resubmitting a query. Therefore, it is promising if we use user query and click histories to improve future search. Another potentially useful idea is to provide users with convenient ways for reviewing search histories.

7 Predicting Query Performance for Personalized Web Search

We compare the performance of each two algorithms and show the results in Table 6. The number in a cell indicates on

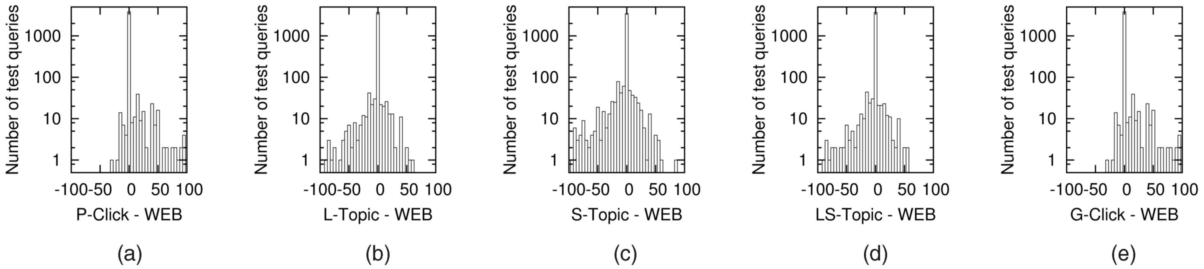


Fig. 5. Distributions of rank scoring increment over the WEB method. The number of test queries with the same rank scoring increment range is plotted in y-axis with log scale. (a) P-Click. (b) L-Topic. (c) S-Topic. (d) LS-Topic $(\theta = 0.5, t = 0.8)$. (e) G-Click $(K = 50)$.

TABLE 6
Performance Comparisons of Each Two Strategies

	WEB	P-C.	L-T.	S-T.	LS-T.	G-C.
WEB	0	30	208	417	204	37
P-C.	148	0	308	516	306	19
L-T.	129	108	0	263	51	112
S-T.	224	205	155	0	130	208
LS-T.	126	109	52	256	0	113
G-C.	144	7	304	511	302	0

how many queries the method given in the row outperforms the method given in the column. For example, the number in the "WEB" row and the "P-C." column indicates that the WEB method outperforms the P-Click method for 30 queries. Please note that the queries on which the two methods achieve the same performance are excluded. Based on this table, we find the following results: 1) The "WEB" row shows that personalization methods do harm search performance for some queries when they improve performance for some other queries. For example, method S-Topic improves search performance for 224 queries, which is even more than those of click-based methods, but it lowers performance for 417 queries. That is why the overall performance of S-Topic is worse than the baseline. If a query set only contains personalization-favored queries, evaluation results will be biased and not trustable to conclude anything on effectiveness. 2) No methods can outperform others for all queries. Different methods have different strengths and weaknesses. Although click-based methods outperform topic-interest-based ones on the average, topic interest-based methods perform best for some queries. If we could combine strengths of these personalization methods intelligently, a better ranking may be achieved in a real-world personalized search engine.

Table 6 shows that personalized search does not perform consistently well under all situations, and it may harm search performance sometimes. It is preferable that we use one personalization algorithm only when it is effective and reduce its harm to search. In this section, we will propose click entropy to measure the variation in user information needs and further build models to predict when a query will benefit from a specific personalization algorithm.

7.1 Click Entropy

As found in [7], personalization may be ineffective for queries that are shown less variation among individuals. In this paper, we define click entropy to measure the variation in user information needs for a query q as follows:

$$ClickEntroy(q) = \sum_{p \in \mathcal{P}(q)} -P(p|q) \log_2 P(p|q). \qquad (7)$$

Here, $ClickEntroy(q)$ is the click entropy of query q. $\mathcal{P}(q)$ is the collection of web pages clicked on query q. $P(p|q)$ is the percentage of clicks on web page p among all clicks on q, i.e.,

$$P(p|q) = \frac{|Clicks(q, p, \bullet)|}{|Clicks(q, \bullet, \bullet)|}.$$

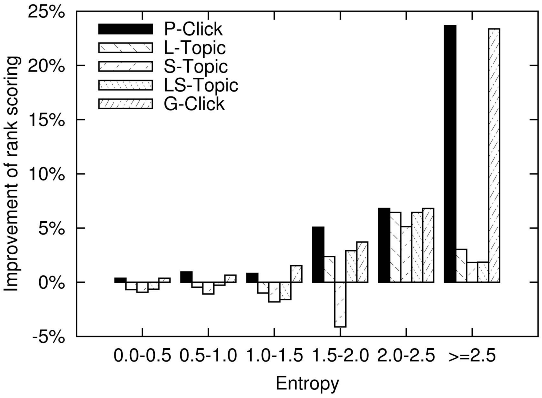


Fig. 6. Search accuracy improvements over the WEB method on the queries with variant click entropies. Only the queries asked by at least three users are included.

Click entropy is a direct indication of query click variation. If all users click only one identical page on query q, we have $ClickEntroy(q) = 0$. A smaller click entropy means that the majority of users agree with each other on a small number of web pages. In such cases, there is no need to do any personalization. A large click entropy indicates that many web pages were clicked for the query. This may mean the following: 1) A user has to select several pages to satisfy his information need, which means that the query is most likely an informational query [17], [18]. In this case, personalization can help to filter the pages that are more relevant to users by making use of historical selections. 2) Different users have different selections on this query, which means that the query is an ambiguous query. In this case, personalization can be used to provide different web pages to different users.

We show the average search performance improvement of different personalization strategies on test queries with different click entropies in Fig. 6. Only the queries that were asked by at least three users are considered.

Fig. 6 shows that the improvement of personalized search performance increases with click entropy being larger, especially when click entropy $\geq$ 1.5. For click-based methods P-Click and G-Click, the improvement is limited for the queries with click entropy being between 0 and 0.5. In the case of a low click entropy, even the best performer, method G-Click, has only 0.37 percent improvement, which is not statistically significant. It indicates that users have general consistent clicks for the queries with a low click entropy, and thus, no personalization is necessary for the current search results. On the contrary, for the queries where click entropy $\geq$ 2.5, the result is obviously different. Both P-Click and G-Click methods make a dramatic improvement. In terms of rank scoring, method G-Click significantly ($p < 0.01$) improves method WEB by 23.37 percent and the P-Click method by 23.68 percent. Topical-interest-based methods L-Topic, S-Topic, and LS-Topic worsen search performance when click entropy $<$ 1.5, whereas L-Topic and LS-Topic achieve better performance for queries with click entropy $\geq$ 1.5. It indicates that queries with a higher click entropy benefit more from personalization.

7.2 Features Used to Predict Query Performance

In Section 7.1, we showed that the proposed click entropy can be used as a simple measurement on whether a query should be personalized. In this section, we will introduce more features that will be used to predict query performance for personalized Web search. Most features are generated based on click-through data, and they require the query to have been issued before (*please note that we omit the query symbol in the remaining parts of this section*). In our previous work [55], we have proposed several features based on search results to predict query ambiguity. Those features would also be helpful to predict query performance (especially for previously unseen queries). The use of search-result-based features and the combination of all types of features are interesting problems themselves. Due to space limitations, we only focus on click-through-based features in this paper and will give deep analysis in future work.

7.2.1 Click Diversity

The goal of personalized Web search is to return different results to different users according to their preferences. A direct way to identify whether users have different preferences on a query is to check the click diversity of users. Click entropy, used in Section 7.1, is one of such measures of click diversity. In this section, we further extract several click-diversity-based features.

For a given query, suppose there are K users who ever issued this query, and there are M documents that are clicked for this query. We calculate click frequencies for each user on each document and represent them in a $K \times M$ user-document matrix $\mathbf{X}$. Each element $x_{k,m} = c$ indicates that user k clicked document m by c times. If the user has not clicked the document, then $x_{k,m} = 0$. Based on this matrix, we extract the following features:

1. **ClickEntropy**. We calculate a probability vector $\mathbf{p}$ based on matrix $\mathbf{P}$ to represent the aggregated click probability distribution over documents:

$$\mathbf{p} = [p_1, \ldots, p_M],\ p_m = \frac{\sum_{i=0}^{K} x_{i,m}}{\sum_{i=0}^{K} \sum_{j=0}^{M} x_{i,j}},\ m = 1, \ldots, M,$$

   where p_m is the probability that users clicked document m on the query. Obviously, $\sum_{m=0}^{M} p_m = 1$. We then calculate **ClickEntropy** based on $\mathbf{p}$ using (7).
2. **ClkProbMean**. We sort the documents by their click probability and then measure how skewed the reordered distribution is. Several standard statistical measurements can be used for this purpose, including the mean, median, skewness, etc. In this paper, we use the mean of the distribution and name this feature as **ClkProbMean**.
3. **UserEntropy**. We normalize matrix $\mathbf{X}$ and get a new matrix $\mathbf{P^u}$, in which each element $p^u_{k,m}$ is computed by

$$p^u_{k,m} = \frac{x_{k,m}}{\sum_{i=0}^{M} x_{k,i}}.$$

   Element $p^u_{k,m} = r$ indicates that user k clicked document m with probability r. $\mathbf{P^u}$ can be decomposed into row vectors:

$$\mathbf{P^u} = [\mathbf{u_1}, \ldots, \mathbf{u_K}]^T,\ \mathbf{u_k} = [p_{k,1}, \ldots, p_{k,M}]^T,\ k = 1, \ldots, K,$$

   where T denotes transpose. Each row vector $\mathbf{u_k^T}$ is a probability vector that corresponds to the user click distribution made by user k on the query. We calculate the average user click entropy, **UserEntropy**, based on these vectors:

$$\begin{aligned} UserEntropy &= \frac{1}{K} \sum_{k=1}^{K} ClickEntropy(\mathbf{u_k}) \\ &= \frac{1}{K} \sum_{i=1}^{K} \sum_{m=0}^{M} -p_{i,m} \log_2 p_{i,m}. \end{aligned}$$

4. **UserDMean**. We calculate the distance between two user click vectors based on the Jensen-Shannon divergence:

$$Dist\left(\mathbf{u_i^T}, \mathbf{u_j^T}\right) = JSD\left(\mathbf{u_i^T} \| \mathbf{u_j^T}\right).$$

   We denote the set of these distances with $\mathbb{D}$:

$$\mathbb{D} = \left\{ Dist\left(\mathbf{u_i^T}, \mathbf{u_j^T}\right) | i \in [1, K], j \in [1, K], i > j \right\}.$$

   We use mean of distances in $\mathbb{D}$ to measure the diversity of user clicks. We denote this feature with **UserDMean**.

7.2.2 Concept Diversity

Another way to identify the diversity of user preferences over a query is to measure the concept/topic diversity of clicked documents. Each document can be classified into one or more concept/topic categories. We suppose that there are N concept categories in the corpus. We use a document-concept matrix $\mathbf{S}_{M\times N}$ to represent categories of documents. Each element $s_{m,n} = f$ indicates that document m would be categorized into concept n with confidence f ($f \in [0, 1]$).

Matrix $\mathbf{S}$ can be decomposed into row vectors:

$$\mathbf{S} = [\mathbf{d_1}, \ldots, \mathbf{d_M}]^T,\ \mathbf{d_m} = [s_{m,1}, \ldots, s_{m,N}]^T,\ m = 1, \ldots, M.$$

Each row vector $\mathbf{d_m^T}$ indicates the classification confidences for document m, and

$$\left|\mathbf{d_m^T}\right| = \sum_{i=0}^{N} s_{m,i} = 1.$$

We use the same concept categories and text classifier as that in Section 4.1.2. We compute the confidence $s_{m,i}$ that document m belongs to ith category as the following equation to make sure that $\sum_{i=0}^{N} s_{m,i} = 1$:

$$s_{m,i} = s^o_{m,i} + \frac{1}{N}\left(1 - \sum_{j=0}^{N} s^o_{m,j}\right),$$

where $s^o_{m,i}$ is the original confidence returned by the classifier. If category i is not in the returned categories, then $s^o_{m,i} = 0$.

We aggregate these concept distribution vectors and calculate entropy based on the normalized concept distribution vector. We name this feature as **CatDEntropy**. We also calculate distances between document-concept vectors and use the mean of these distances to measure the concept diversity of queries. We denote this feature with **CatDMean**.

We can also incorporate document click probability vector $\mathbf{p}$ into matrix $\mathbf{S}$ and generate a new matrix $\mathbf{S^w}$:

$$\mathbf{S^w} = \left[\mathbf{d_1^w}, \ldots, \mathbf{d_M^w}\right]^T = [p_1 * \mathbf{d_1}, \ldots, p_m * \mathbf{d_M}]^T,\ m = 1, \ldots, M.$$

Then, a feature, **WeightedCatDMean,** is extracted from $\mathbf{S^w}$ using a similar method to **CatDMean**.

7.2.3 User Concept Diversity

We compute the product of matrices $\mathbf{P_{K\times M}}$ and $\mathbf{S_{M\times N}}$ and get a user-concept user profile matrix $\mathbf{G_{K\times N}}$ for this query:

$$g_{k,n} = \sum_{i=0}^{M} p_{k,i} * s_{i,n}.$$

$g_{k,n}$ is the probability that user k selected a document that belongs to subject n.

Matrix $\mathbf{G}$ can be decomposed into row vectors:

$$\mathbf{G} = [\mathbf{a_1}, \ldots, \mathbf{a_K}]^T, \mathbf{a_k} = [g_{k,1}, \ldots, g_{k,N}]^T,\ k = 1, \ldots, K.$$

Each row vector $\mathbf{a_k^T}$ corresponds to user concept preferences for the query. Then, a similar feature, **UserCatDMean**, is extracted from the set of vectors $\mathbf{a_k^T}$.

7.2.4 Other Features

We also extract some simple features:

1. **ExRatio**. By observing click-through data, we find that an ambiguous query is usually reformulated by users. A common reformulation is adding terms to the original query. So, we extract feature ExRatio based on this information. Suppose NumOfSessions is the number of sessions that the query appears and NumOfSessionsEx is the number of sessions that the query appears and at least one extended query also appears. An extended query q_x of query q either starts or ends with q. ExRatio is calculated by **ExRatio** $= \mathrm{NumOfSessionsEx}/\mathrm{NumOfSessions}$.
2. **IsFirstQueryInSession**. Obviously, if a query is the first query of a session, S-Topic cannot work for it. This feature will be useful for identifying whether to use S-Topic.
3. **HasQueryHistory**. This feature indicates whether the query has been issued in the past. It is useful for P-Click and G-Click methods.
4. **AvgClkTimes**. This feature is the average historical click times per query for the query string. If users usually click multiple results for a query, this query is more likely to be an ambiguous or informational query.

7.2.5 Discussion about Features

Most proposed features are extracted from click-through data. Similar to the data processes introduced in Section 4.3, we need to prepare grouped click-through data and concept vectors of pages in advance. Features can be calculated either online or offline. For popular queries that are issued by a large numbers of users, we prefer to generate feature offline because these queries may be frequently issued by users. Furthermore, feature UserDMean and UserCatDMean may bring high computation cost when the number of users is very large.

7.3 Personalization-Faced Query Classification

In this section, we use the proposed features to learn query classification models to predict whether a query should be personalized using a specific personalization algorithm. Table 3 shows that topical-interest-based strategies perform worse than baseline ranking. Table 6 shows that they improve ranking accuracy for many queries, but they also harm performance on more queries. If we can successfully predict when these algorithms would improve ranking and when they would harm ranking, we can use them only when they are helpful and hence reduce its harm to performance. Due to space limitations, we only discuss the typical topical-interest-based method L-Topic and leave other two strategies to future work. We do not explore query classification for click-based methods that perform better than topical-interest-based ones in this paper. Table 6 shows that only a small number of queries are harmed by these methods, and therefore, only a small improvement will be achieved if we apply query classification for them.

TABLE 7
Accuracy of the Prediction Model

	WEB	L-Topic	Macro Average
Precision	0.7678	0.8917	0.8297
Recall	0.9890	0.2218	0.6054
F1	0.8644	0.3534	0.6994
Accuracy	0.7759	0.7759	0.7759

In Section 6, we got rank scoring values of the L-Profile algorithm for each test query. We label the types of test queries based on these values. The queries on which L-Topic outperforms the nonpersonalization method (WEB) are assigned a label **L-Topic**, and those on which L-Topic harms search accuracy are labeled as **WEB**. Please note that we simply remove the queries on which methods L-Profile and WEB perform equally well. Finally, 364 queries are labeled as WEB, and 140 queries are labeled as L-Topic. We use all 504 queries as our data set in this section.

We use a support vector machine (SVM) classifier [56] with RBF kernel to accomplish the classification task and use eightfold cross validation to tune parameters. We purposively select parameters to produce high precision for the L-Topic class because we want to reduce the harm of L-Topic. High precision for the L-Topic class guarantees that most queries predicted as L-Topic can be successfully personalized by L-Topic. The classification results are shown in Table 7. This table shows that we can successfully predict query types for about 78 percent of test queries. Class L-Topic has very high precision but low recall. We will investigate more effective features and increase the recall of class L-Topic in future work. According to the predicted results given in each test fold, we use the following rule to decide whether to use L-Topic on each test query. If the type of a query has been predicted as L-Topic, we use the L-Topic algorithm to personalize it. If its type has been predicted as WEB, we use the original ranking for it. We denote this hybrid strategy as **SelPer**. Please note that for test queries on which method L-Profile performs equally to WEB, we can select any method (L-Topic or WEB) because L-Topic neither harms nor improves performance for these queries. Experimental results show that the SelPer algorithm yields a ranking score of 70.49. It outperforms WEB and L-Topic by 1.47 percent and 2 percent, respectively (both improvements are significant with $p < 0.01$). That means that using the L-Topic algorithm only for selected queries is better than using it simply for all queries, and our prediction model is therefore helpful to personalized Web search.

8 Conclusions

In this paper, we developed an evaluation framework based on real query logs to enable large-scale evaluation of personalized search. We used 12 days of Windows Live query logs to evaluate five personalized search algorithms. In the experiments, click-based personalization algorithms worked well. Although the algorithms work only for repeated queries, they are simple and stable. We suggest that search engines take advantage of user histories in search if privacy issues do not prohibit it. The topical-interest-based personalized search algorithms implemented were not as stable as the click-based ones under our framework (because of the presentation bias, this conclusion is less strong). They could improve search accuracy for some queries, but they harmed performance for more queries. As these methods were not optimal, we will continue our evaluation work on improved versions in the future.

In most previous work on personalized Web search, all queries were usually personalized in the same manner. Another important conclusion we revealed in this paper is that personalization does not work equally well under various situations. We defined the click entropy to measure variation in information needs of users under a query. Experimental results showed that personalized Web search yields significant improvements over generic Web search for queries with a high click entropy. For the queries with a low click entropy, personalization methods performed similarly or even worse than generic search. As personalized search had different effectiveness for different kinds of queries, we argued that queries should not be handled in the same manner with regard to personalization. Our proposed click entropy can be used as a simple measurement on whether a query should be personalized.

We further proposed several features to automatically predict whether a query can benefit from a specific personalization algorithm. Due to space limitations, we only experimented with the L-Topic algorithm. Experimental results showed that using the L-Topic algorithm for queries selected by our prediction model would achieve better overall performance than using it simply for all queries. We will try to build prediction models for other algorithms in future work.

We found that no personalization algorithms can outperform others for all queries. Different methods have different strengths and weaknesses. A promising direction we will explore in the future is to automatically predict which algorithm should be used for given a query and/or to combine the strengths of different personalization methods.

Acknowledgments

The authors are grateful to Dwight Daniels and Matthew Callcut for the edits and comments on writing the paper.

References

[1] C. Silverstein, H. Marais, M. Henzinger, and M. Moricz, "Analysis of a Very Large Web Search Engine Query Log," *ACM SIGIR Forum,* vol. 33, no. 1, pp. 6-12, 1999.

[2] B.J. Jansen, A. Spink, and T. Saracevic, "Real Life, Real Users, and Real Needs: A Study and Analysis of User Queries on the Web," *Information Processing and Management,* vol. 36, no. 2, pp. 207-227, 2000.

[3] R. Krovetz and W.B. Croft, "Lexical Ambiguity and Information Retrieval," *Information Systems,* vol. 10, no. 2, pp. 115-141, 1992.

[4] S. Cronen-Townsend and W.B. Croft, "Quantifying Query Ambiguity," *Proc. Second Int'l Conf. Human Language Technology Research (HLT '02),* pp. 94-98, 2002.

[5] X. Shen, B. Tan, and C. Zhai, "Implicit User Modeling for Personalized Search," *Proc. ACM Int'l Conf. Information and Knowledge Management (CIKM '05),* pp. 824-831, 2005.

[6] F. Qiu and J. Cho, "Automatic Identification of User Interest for Personalized Search," *Proc. 15th Int'l World Wide Web Conf. (WWW '06),* pp. 727-736, 2006.

[7] J. Teevan, S.T. Dumais, and E. Horvitz, "Beyond the Commons: Investigating the Value of Personalizing Web Search," *Proc. Workshop New Technologies for Personalized Information Access (PIA),* 2005.

[8] J. Pitkow, H. Schutze, T. Cass, R. Cooley, D. Turnbull, A. Edmonds, E. Adar, and T. Breuel, "Personalized Search," *Comm. ACM,* vol. 45, no. 9, pp. 50-55, 2002.

[9] A. Pretschner and S. Gauch, "Ontology Based Personalized Search," *Proc. 11th IEEE Int'l Conf. Tools with Artificial Intelligence (ICTAI '99),* pp. 391-398, 1999.

[10] B. Tan, X. Shen, and C. Zhai, "Mining Long-Term Search History to Improve Search Accuracy," *Proc. 12fth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD '06),* pp. 718-723, 2006.

[11] G. Jeh and J. Widom, "Scaling Personalized Web Search," *Proc. 12th Int'l World Wide Web Conf. (WWW '03),* pp. 271-279, 2003.

[12] P. Ferragina and A. Gulli, "A Personalized Search Engine Based on Web-Snippet Hierarchical Clustering," *Special Interest Tracks and Posters of the 14th Int'l Conf. World Wide Web (WWW '05),* pp. 801-810, 2005.

[13] J. Teevan, S.T. Dumais, and E. Horvitz, "Personalizing Search via Automated Analysis of Interests and Activities," *Proc. 28th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '05),* pp. 449-456, 2005.

[14] J.-T. Sun, H.-J. Zeng, H. Liu, Y. Lu, and Z. Chen, "CubeSVD: A Novel Approach to Personalized Web Search," *Proc. 14th Int'l World Wide Web Conf. (WWW '05),* pp. 382-390, 2005.

[15] F. Liu, C. Yu, and W. Meng, "Personalized Web Search by Mapping User Queries to Categories," *Proc. ACM Int'l Conf. Information and Knowledge Management (CIKM '02),* pp. 558-565, 2002.

[16] P.-A. Chirita, W. Nejdl, R. Paiu, and C. Kohlschütter, "Using ODP Metadata to Personalize Search," *Proc. 28th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '05),* pp. 178-185, 2005.

[17] A. Broder, "A Taxonomy of Web Search," *ACM SIGIR Forum,* vol. 36, no. 2, pp. 3-10, 2002.

[18] U. Lee, Z. Liu, and J. Cho, "Automatic Identification of User Goals in Web Search," *Proc. 14th Int'l World Wide Web Conf. (WWW '05),* pp. 391-400, 2005.

[19] X. Shen, B. Tan, and C. Zhai, "Context-Sensitive Information Retrieval Using Implicit Feedback," *Proc. 28th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '05),* pp. 43-50, 2005.

[20] Windows Live Search, http://www.live.com, 2006.

[21] J.-R. Wen, Z. Dou, and R. Song, "Personalized Web Search," *Encyclopedia of Database Systems,* 2009.

[22] J.M. Carroll and M.B. Rosson, "Paradox of the Active User," *Interfacing Thought: Cognitive Aspects of Human-Computer Interaction,* pp. 80-111, 1987.

[23] K. Sugiyama, K. Hatano, and M. Yoshikawa, "Adaptive Web Search Based on User Profile Constructed without Any Effort from Users," *Proc. 13th Int'l World Wide Web Conf. (WWW '04),* pp. 675-684, 2004.

[24] F. Liu, C. Yu, and W. Meng, "Personalized Web Search for Improving Retrieval Effectiveness," *IEEE Trans. Knowledge and Data Eng.,* vol. 16, no. 1, pp. 28-40, Jan. 2004.

[25] P.A. Chirita, C. Firan, and W. Nejdl, "Summarizing Local Context to Personalize Global Web Search," *Proc. ACM Int'l Conf. Information and Knowledge Management (CIKM),* 2006.

[26] J. Chaffee and S. Gauch, "Personal Ontologies for Web Navigation," *Proc. ACM Int'l Conf. Information and Knowledge Management (CIKM '00),* pp. 227-234, 2000.

[27] S. Gauch, J. Chaffee, and A. Pretschner, "Ontology-Based Personalized Search and Browsing," *Web Intelligence and Agent Systems,* vol. 1, no. 3/4, pp. 219-234, 2003.

[28] J. Trajkova and S. Gauch, "Improving Ontology-Based User Profiles," *Proc. Recherche d'Information Assistée par Ordinateur (RIAO '04),* pp. 380-389, 2004.

[29] M. Speretta and S. Gauch, "Personalized Search Based on User Search Histories," *Proc. IEEE/WIC/ACM Int'l Conf. Web Intelligence (WI '05),* pp. 622-628, 2005.

[30] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank Citation Ranking: Bringing Order to the Web," technical report, Computer Science Dept., Stanford Univ., 1998.

[31] T.H. Haveliwala, "Topic-Sensitive Pagerank," *Proc. 11th Int'l World Wide Web Conf. (WWW),* 2002.

[32] T. Sarlós, A.A. Benczúr, K. Csalogány, D. Fogaras, and B. Rácz, "To Randomize or Not to Randomize: Space Optimal Summaries for Hyperlink Analysis," *Proc. 15th Int'l World Wide Web Conf. (WWW '06),* pp. 297-306, 2006.

[33] F. Tanudjaja and L. Mui, "Persona: A Contextualized and Personalized Web Search," *Proc. 35th Hawaii Int'l Conf. System Sciences (HICSS '02),* vol. 3, p. 53, 2002.

[34] J.S. Breese, D. Heckerman, and C. Kadie, "Empirical Analysis of Predictive Algorithms for Collaborative Filtering," *Proc. 14th Conf. Uncertainty in Artificial Intelligence (UAI '98),* pp. 43-52, 1998.

[35] P.A. Chirita, C.S. Firan, and W. Nejdl, "Personalized Query Expansion for the Web," *Proc. 30th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '07),* pp. 7-14, 2007.

[36] J. Teevan, S.T. Dumais, and D.J. Liebling, "To Personalize or Not to Personalize: Modeling Queries with Variation in User Intent," *Proc. 31th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '08),* 2008.

[37] S. Cronen-Townsend, Y. Zhou, and W.B. Croft, "Predicting Query Performance," *Proc. 25th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '02),* pp. 299-306, 2002.

[38] E. Yom-Tov, S. Fine, D. Carmel, and A. Darlow, "Learning to Estimate Query Difficulty: Including Applications to Missing Content Detection and Distributed Information Retrieval," *Proc. 28th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '05),* pp. 512-519, 2005.

[39] D. Carmel, E. Yom-Tov, A. Darlow, and D. Pelleg, "What Makes a Query Difficult?" *Proc. 29th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '06),* pp. 390-397, 2006.

[40] Y. Zhou and W.B. Croft, "Query Performance Prediction in Web Search Environments," *Proc. 30th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '07),* pp. 543-550, 2007.

[41] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay, "Accurately Interpreting Clickthrough Data as Implicit Feedback," *Proc. 28th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '05),* pp. 154-161, 2005.

[42] Z. Guan and E. Cutrell, "An Eye Tracking Study of the Effect of Target Rank on Web Search," *Proc. ACM SIGCHI Conf. Human Factors in Computing Systems (CHI '07),* pp. 417-420, 2007.

[43] J. Boyan, D. Freitag, and T. Joachims, "Evaluating Retrieval Performance Using Clickthrough Data," *Proc. AAAI Workshop Internet-Based Information Systems,* 1996.

[44] J.C. Borda, "Mémoire sur les Élections au Scrution," *Histoire de l'Académie Royal des Sciences,* 1781.

[45] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar, "Rank Aggregation Methods for the Web," *Proc. 10th Int'l World Wide Web Conf. (WWW '01),* pp. 613-622, 2001.

[46] Y. Li, Z. Zheng, and H.K. Dai, "KDD CUP-2005 Report: Facing a Great Challenge," *ACM SIGKDD Explorations Newsletter,* vol. 7, no. 2, pp. 91-99, 2005.

[47] D. Shen, R. Pan, J.-T. Sun, J.J. Pan, K. Wu, J. Yin, and Q. Yang, "Q2C@UST: Our Winning Solution to Query Classification in KDDCUP 2005," *ACM SIGKDD Explorations Newsletter,* vol. 7, no. 2, pp. 100-110, 2005.

[48] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly, "Dryad: Distributed Data-Parallel Programs from Sequential Building Blocks," *Proc. EuroSys '07,* pp. 59-72, 2007.

[49] W. Lin, M. Yang, L. Zhang, and L. Zhou, "Pacifica: Replication in Log-Based Distributed Storage Systems," Technical Report MSR-TR-2008-25, Micorsoft Research, 2008.

[50] Y. Xie and D.R. O'Hallaron, "Locality in Search Engine Queries and Its Implications for Caching," *Proc. IEEE INFOCOM,* 2002.

[51] B.J. Jansen, A. Spink, J. Bateman, and T. Saracevic, "Real Life Information Retrieval: A Study of User Queries on the Web," *ACM SIGIR Forum,* vol. 32, no. 1, pp. 5-17, 1998.

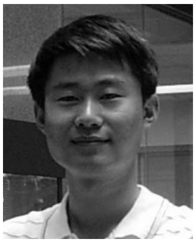
[52] S. Wedig and O. Madani, "A Large-Scale Analysis of Query Logs for Assessing Personalization Opportunities," *Proc. 12th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD '06),* pp. 742-747, 2006.

[53] S.M. Beitzel, E.C. Jensen, A. Chowdhury, D. Grossman, and O. Frieder, "Hourly Analysis of a Very Large Topically Categorized Web Query Log," *Proc. 27th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '04),* pp. 321-328, 2004.

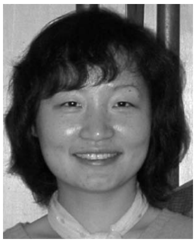
[54] J. Teevan, E. Adar, R. Jones, and M. Potts, "History Repeats Itself: Repeat Queries in Yahoo's Logs," *Proc. ACM SIGIR '06,* pp. 703-704, 2006.

[55] R. Song, Z. Luo, J.-R. Wen, Y. Yu, and H.-W. Hon, "Identifying Ambiguous Queries in Web Search," *Proc. 16th Int'l World Wide Web Conf. (WWW '07),* pp. 1169-1170, 2007.

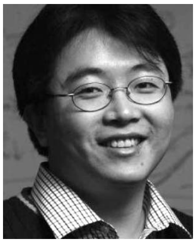
[56] A. Jain and D. Zongker, "Feature Selection: Evaluation, Application, and Small Sample Performance," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 19, no. 2, pp. 153-158, Feb. 1997.



Zhicheng Dou received the BS and PhD degrees in computer science and technology from Nankai University, Tianjin, China, in 2003 and 2008, respectively. He is currently an associate researcher with Microsoft Research Asia, Beijing. His main research interests include personalized Web search, Web information retrieval, data mining, and information systems.



Ruihua Song received the BE and ME degrees from the Department of Computer Science and Technology, Tsinghua University. She is a researcher with Microsoft Research Asia, Beijing. Her main research interests are Web information retrieval and Web information extraction.



Ji-Rong Wen received the BS and MS degrees from Renmin University of China and the PhD degree in 1999 from the Institute of Computing Technology, Chinese Academy of Science. He is currently a research manager with Microsoft Research Asia, Beijing. His main research interests are Web data management, information retrieval (especially Web IR), data mining, and machine learning.

Xiaojie Yuan received the MS degree in computer science and the PhD degree in control engineering from Nankai University, Tianjin, China, in 1988 and 2000, respectively. She is a professor in the Department of Computer Science, Nankai University. She is also a vice dean of the College of Information Technical Science. Her primary research covers integration of heterogeneous data sources, Web information retrieval, XML data management, and software reverse engineering. She is also a member of the China Computer Federation Software Engineering Society.