Enhancing Personalized Search by Mining and Modeling Task Behavior

Ryen W. White[1], Wei Chu[2], Ahmed Hassan[1], Xiaodong He[1], Yang Song[1], Hongning Wang[3]
[1] Microsoft Research, Redmond, WA 98052 USA
[2] Microsoft Bing, Bellevue, WA 98004 USA
[3] Department of Computer Science, University of Illinois at Urbana Champaign, Urbana, IL 61801 USA
{ryenw, wechu, hassanam, xiaohe, yangsong}@microsoft.com, wang296@illinois.edu

ABSTRACT

Personalized search systems tailor search results to the current user intent using historic search interactions. This relies on being able to find pertinent information in that user's search history, which can be challenging for unseen queries and for new search scenarios. Building richer models of users' current and historic search *tasks* can help improve the likelihood of finding relevant content and enhance the relevance and coverage of personalization methods. The task-based approach can be applied to the current user's search history, or as we focus on here, all users' search histories as so-called "groupization" (a variant of personalization whereby other users' profiles can be used to personalize the search experience). We describe a method whereby we mine historic search-engine logs to find other users performing similar tasks to the current user and leverage their on-task behavior to identify Web pages to promote in the current ranking. We investigate the effectiveness of this approach versus query-based matching and finding related historic activity from the current user (i.e., group versus individual). As part of our studies we also explore the use of the on-task behavior of particular user cohorts, such as people who are expert in the topic currently being searched, rather than all other users. Our approach yields promising gains in retrieval performance, and has direct implications for improving personalization in search systems.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval –*search process*, *selection process.*

General Terms

Algorithms, Experimentation, Human Factors

Keywords

Task modeling, Task similarity, Personalization, Groupization.

1. INTRODUCTION

Search engines record queries and search-result clicks from their users and leverage that data to enhance result relevance for others issuing the same or similar queries [1,21]. The underlying motivation behind this query-based matching is to find other users with similar information needs, and use their aggregated search behavior to estimate the current user's underlying intent. Historic search interactions from a user over time can be used to personalize search results [37,42], but the focus there is either once again on query-based matching [42] or creating general models of searcher interests across a variety of topics [37]. However, since queries occur in a broader task context, focusing only on query- or topical-interest-matching may be insufficient for effective search-result ranking.

People have been shown to pursue a wide range of different search tasks online [23,31] and inferences about task behavior have been shown to have value in areas such as modeling search satisfaction [20]. There have also been attempts to leverage the on-task behavior of other users to improve retrieval performance. Research on *groupization* [43] showed that extending personalization to groups of users with shared interests could yield relevance gains. However this used information that is typically unavailable to search engines (e.g., length of members' relationships) and is on a small scale. Collaborative filtering attempts to find others with similar interests [13,25], but matches based only on queries [25] or focuses on recommendations and community connectedness not ranking [13].

We believe that by directly modeling task-relevant search behavior as part of personalization we can attain improved search result relevance. In this particular study, we are interested in first modeling users' on-task search behavior, then using the generated task model to find other users attempting similar tasks, identifying the URLs that appear relevant, and then promoting those URLs in the result list for the current query. We answer four questions critical in determining the value of this method: (1) Should matching be performed using task models or is finding other instances of the current query sufficient? (2) How does task-based groupization perform in comparison with task-based personalization? (3) Is in-session task segmentation required to attain performance gains or would an estimation of tasks via search sessions suffice? (4) What is the effect of using specific user cohorts for groupization (e.g., those in a particular geographic location or those with good topic knowledge)? Through empirical analysis we demonstrate that mining and modeling search tasks yields significant gains in retrieval performance.

The remainder of this paper is structured as follows. Section 2 describes related work in task modeling, personalization, and mining task-relevant search behavior. Sections 3 and 4 describe the identification of search tasks and the method for the learning to rank search results by using evidence from similar tasks. Section 5 describes the experiments that we performed to evaluate our methods and Section 6 describes the results. In Section 7 we discuss these findings and their implications, and we conclude in Section 8.

2. RELATED WORK

There are three relevant areas of related work: (1) task modeling, (2) personalization of search engines based on short- and long-term searcher interests, and (3) mining the search behavior of other users to complement and enhance search personalization.

An important part of representing search intent is understanding the various types of search tasks and the different motivations that searchers may have for pursuing their information goals. Previous work has studied the motivation for searching and nature of the search tasks that people perform [14,26]. There has also been research on mining and modeling task-related search behavior from search logs. Jones and Klinkner [22] proposed to classify the query pairs that belong to the same task using, among others, features of

WWW 2013, May 13–17, 2013, Rio de Janiero, Brazil.
ACM 978-1-4503-2035-1/13/05.

query-term overlap and search result similarity. They showed in experiments that their approach attained significant (over 90%) accuracy in segmenting and matching search tasks. Such task models have also been used to predict search success automatically from observed search behavior [18,19,20]. Lucchese et al. [30] identified task-based sessions by combining content (query term edit distance) and semantic (Wikipedia) features. Liao et al. [27] adapt both of the methods described in [19,30] to extract tasks from sessions where a query distance function is learned and used to cluster queries in sessions into tasks. They studied the potential benefit of using tasks for search applications—determining user satisfaction, predicting search interests, and query suggestion—and demonstrate benefit from tasks over sessions or queries. Others have modeled cross-session interests to predict task continuation and resumption [24] or have sought to understand multi-session tasks [31].

Large-scale behavioral data from search engines has been mined extensively to improve search result relevance [1,21]. Radlinski and Joachims [33] proposed the use of query chains comprising connected sequences of queries to learn richer models of relevance that can capitalize on session behavior. However, the basis for the learning is still individual queries, which may not map well to the current user's task. Moving beyond individual queries, Radlinski et al. [34] model intent from queries and clicks in a way that could be directly consumed by Web search engines. Similarly, Downey et al. [12] studied relationships between queries and goals, estimated from the terminal page in the search session. Bilenko and White [7] used signals from aggregated post-query navigation trails to learn better result rankings using search behavior.

Research on personalizing retrieval [35,41] has found that implicitly gathered information such as browser history, query history, and desktop information, can be used to improve the ranking of search results for a given user. Short-term behavior from within the current search session has been used for tasks such as search result ranking [49] or predicting future search interests [45,46]. Teevan et al. [41] found that the performance of the personalization algorithm they studied was improved as more data became available about the target user's interests. Long-term behavior has been used for personalizing search result ranking by building long-terms models of search interests [37], including specifically using previous queries suggesting a pursuit of similar information needs [40]. Other work has focused on personalization based on task-type, including the connection between task type and search behavior [29], and using those signals for personalization [28], although within the same user rather than over many users as we focus on in our method.

When there are insufficient data about the current user, the search behavior of other related users may be beneficial. Teevan et al. [43] explored the similarity of queries, desktop information, and explicit relevance judgments across a small group of 30 work colleagues grouped along two dimensions: (1) the longevity of their relationships, and (2) how explicitly the group is formed. They found that some groupings provide insight into what members consider relevant to queries related to the group focus, but that it can be difficult to identify valuable groups implicitly. We address this challenge directly in this paper, and experiment with different methods for identifying groups to enhance personalization performance at scale.

Collaborative filtering (CF) can be used to find people with similar interests and leverage their activities and preferences to help the current user. There are examples of CF techniques being applied to improve search. Sugiyama et al. [38] addressed sparseness in user term-weight profiles by applying collaborative filtering techniques to provide term weights based on those of users with similar profiles. Similar approaches have used clickthrough data to personalize result rankings and backed off to the clicks of other users [2,39]. Almeida and Almeida [2] used Bayesian algorithms to cluster users of an online bookstore's search service into communities based on links clicked within the site and found that the popularity of links within different communities could be used to customize result rankings. Lee [25] used data mining to uncover patterns in users' queries and browsing to generate recommendations for users with similar queries. All techniques perform matching with other users based on individual queries or URLs, severely limiting their coverage. One way to address this is to use clickthrough behavior. Freyne and Smyth [13] tried to connect different communities based on the degree to which communities' queries and result clicks overlap.

Other methods have been proposed that are query independent. Smyth [36] suggested that clickthrough data from users in the same "search community" (e.g., a group of people who use a special-interest Web portal or who work at the same company) could enhance search result lists. Smyth provided evidence for the existence of search communities by showing that a group of employees from a single company had a higher query similarity threshold than general Web users. Mei and Church [32] found that geographic location might serve as a reasonable proxy for community, since they observed that grouping users based on the similarity of their IP addresses could improve search results. As part of the research presented in this paper, we study the utility of location-based cohorts.

Our research extends previous work in the following ways. First, we model users' tasks and learn from their task-related behavior rather than only using what they do for individual queries or general topical interests. Second, rather than personalizing using the user's own on-task behavior we have developed methods to leverage task models from other users attempting similar tasks. As part of that aspect of our study, we compare the retrieval performance of task-based groupization with task-based personalization. Third, we address the scalability challenges of implicitly modeling task similarity and show gains from leveraging groupization at scale. Finally, we study the effectiveness of using the activities of different cohorts based on location, browser / entry point denoting how users reach the search engine, and high levels of domain expertise.

3. IDENTIFYING TASKS

The first step in applying our method is to identify tasks within search sessions. We now describe the task identification process.

3.1 Log Data

The primary source of data for this study is a proprietary data set comprising anonymized logs of users of the Microsoft Bing search engine. The logs contained a unique user identifier, a search session identifier, the query, the top-10 URLs returned by the search engine for that query, and clicks on the results. We used four weeks of log data gathered from July 2011 to generate features, and to train and evaluate our different ranking models. Logs were collected during A/B tests where other types of personalization support was disabled, so as to not bias our results with other personalization signals. Logs were split into search sessions demarcated with a 30-minute inactivity timeout, such as that used in previous work [12,33].

3.2 Building Task Models

3.2.1 Identifying Tasks in Sessions

To extract tasks from within search sessions, we use the query clustering method QTC [27], which has the advantage of segmenting interleaved tasks within a session. The method works in two steps: first, measure the similarities between query pairs; second, cluster queries into tasks based on their similarity scores.

In order to calculate inter-query similarities, QTC takes a supervised-learning approach. First, human assessors are asked to assign binary labels to a set of randomly-sampled query pairs. A query pair has a positive label if assessors think they are related, e.g., repeated queries [amazon] → [amazon], one query contains narrowing intent of the other [disney] → [disney movies], etc. Otherwise they are assigned a negative label if the queries are unrelated or contain different atomic intentions, e.g., [seattle news] → [space needle]. A logistic regression classifier is then trained with a set of term and temporal features based on the human labels.

Using the learned query similarity function, QTC then builds an undirected graph of queries within each user session, where the vertices of the graph are queries and the edges represent similarities between queries. By dropping the weak edges where the similarities are smaller than a threshold which is determined using cross validation (0.5 in our case), we can extract all connected components of the graph as tasks. See [27] for more on QTC.

3.2.2 Modeling Search Tasks

Now that we have a way to identify the search tasks within sessions, we need to represent searchers' tasks in a way that enables comparisons between them. The two search behaviors that are readily available to us in the logs are queries and search-result clicks. We leverage these two sources to build the following four representations of users' on-task behavior: queries, clicked-result URLs, the Web domains of the clicked results, and topical labels for clicked results from the Open Directory Project (ODP, dmoz.org). Using these four sources, tasks are represented as both sets (for queries, query terms, clicked URLs, clicked URL domains), and as probability distributions across topical category labels assigned to the URLs. For clicked URLs, we only used those with a dwell time exceeding 30 seconds, suggesting that the user was satisfied [14].

The topical labels from ODP were assigned in an automated manner to all URLs in the Bing index using a content-based classifier, described and evaluated in [4]. In turn, this provided us with category information for all search-result clicks. The classifier employs logistic regression to predict the ODP category for a Web page. To lessen the impact of small differences in assigned labels, we use only 219 categories at the top two levels of the ODP hierarchy. The findings in [4] revealed that, when optimized for the score in each category, the content-based classifier has a micro-average F1 of 0.60, which we believed was sufficient for our purposes.

The sources chosen were all available to us at scale and allowed us to compute task similarity along a number of different dimensions. The inter-task similarity features that leverage these representations are described later. Before we discuss these features and how we learn from similar tasks, we present some brief summary statistics on the characteristics of search tasks that we mined from the logs.

3.3 Task Characteristics

In the one-week of log data (from July 1, 2012 to July 8, 2012) used later for feature generation there were more than three million impressions, 1.4 million search sessions, and 1.9 million search tasks. This represented an average of 1.36 tasks per session, 2.52 queries per session, and 1.86 queries per task. Figure 1 illustrates the fraction of sessions containing between one and five search tasks. The figure shows that around 90% of sessions have one or two tasks; 73.3% sessions contain a single task and about 16.0% sessions contain two tasks. This shows that although most sessions comprise a single task, there are still a sizable number of sessions (over 25%) containing multiple tasks. Since using all in-session activity may result in a noisier relevance signal, we may need to consider in-session task boundaries. We explore the effect of using full-session search activity versus in-session task activity as part of the ranking experiments described later in the paper.

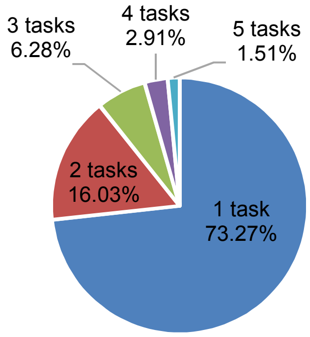

Figure 1. Number of search tasks in search sessions.

We now describe the process by which we learn from similar tasks.

4. LEARNING FROM SIMILAR TASKS

Given a user attempting a search task, the goal of our method is to learn from the on-task search behavior of other users. A key part of this process is finding other users attempting the same or similar search tasks. In this section we describe the methods that we use to compute the similarity between pairs of search tasks, how we mine similar tasks, and the features that we generate for ranking.

4.1 Computing Task Similarity

There were a number of ways in which we computed the similarity between a given pair of tasks. These can be grouped together as two similarity features classes: query similarity and result similarity.

4.1.1 Query Similarity

These similarity measures are based on comparing the queries that users issue in both tasks under consideration. Similarity in this case can be based on the exact terminology used in the queries (after normalization) and more generally, on the semantic similarity between the queries. We consider each of these alternatives.

4.1.1.1 Syntactic Similarity

Syntactic similarity describes the string match between the queries. Similarity can be computed based on the overlap between the tasks in terms of: (1) the fraction of queries that are shared between tasks (i.e., the intersection divided by the union), and (2) as the fraction of unique query terms that are shared between tasks.

4.1.1.2 Semantic Similarity

While the queries in two tasks may not have direct term overlap, they may be similar semantically. To address this we also compute the task similarity by measuring the semantic similarity between queries of two tasks. Let $Q = q_1 \dots q_J$ be one query and $S = s_1 \dots s_I$ be another, the semantic similarity between these two queries can be measured based on the IBM Model 1 [6,8]. IBM Model 1 was originally proposed to model the probability of translating from one sequence of words in one language to another. Later, the model was applied to various information retrieval (IR) tasks such as query expansion [16] and document ranking [15]. Treating Q and S as two sequences of words, the IBM Model 1-based semantic similarity model is defined as:

$$P(S|Q) = \prod_{i=1}^{I} \sum_{j=1}^{J} P(s_i|q_j) P(q_j|Q) \quad (1)$$

where $P(q|Q)$ is the unigram probability of word q in query Q. The word translation probabilities $P(s|q)$ are estimated on the query-title pairs derived from the clickthrough search logs, assuming that the title terms are likely to be the desired alternation of the paired query. Our method follows the standard training procedure of IBM model 1 as proposed by Brown et al. [8]. Formally, we optimize the model parameters θ, i.e., the set of all word translation probabilities $\{P(s|q)\}$, by maximizing the probability of generating document titles from queries over the entire training corpus:

$$\theta^* = \mathrm{argmax}_\theta \prod_{i=1}^{H} P(S_i|Q_i, \theta) \qquad (2)$$

where $P(S|Q, \theta)$ takes the form of IBM Model 1:

$$P(S|Q, \theta) = \frac{\varepsilon}{(J+1)^I} \prod_{i=1}^{I} \sum_{j=1}^{J} P(s_i|q_j) \qquad (3)$$

where ε is a constant, I is the token length of S, and J is the token length of Q. The query-title pairs used for model training are sampled from one year of search logs from the Bing search engine, without any overlap with the experiments reported in this paper.

We compute the sematic similarity between two tasks using the average $P(S|Q, \theta)$ across all pairs of queries from the tasks.

4.1.2 Clicked-Result Similarity

The result URLs that are clicked (and have an associated long dwell) provide a different source of information about users' search intent than is available in queries. We compute similarity in three ways: (1) the match between the clicked URLs, (2) the match between the domains of the clicked URLs, and even more generally, (3) the match between the topical categories assigned to the URLs.

4.1.2.1 URL and Domain Similarity

Similarity is computed based on the overlap between the tasks in terms of the fraction of unique clicked URLs shared between them (i.e., the intersection of clicked URLs divided by the union of all clicked URLs), or in terms of the URL domains that are shared between the tasks. Backing off to Web domains provides more opportunity for a match between clicked results, improving coverage while preserving information about the web site of interest.

4.1.2.2 Topical Similarity

Rather than relying on exact matches between particular URLs or their domains, we can also consider matching based on the topicality of the pages described earlier. Given a categorization for each of the clicked URLs for a task t, we can create an ODP category distribution C_t for that task, with a probability for each topical category c (i.e., $P_t(c)$). Given this representation, we can compare the distribution for the current task with the distribution over all other tasks from other users $\{t'\}$. We perform this comparison using both Kullback-Liebler divergence and the cosine similarity. That is,

$$KL(t', t) = \sum_{c \in C} \ln\left(\frac{P_{t'}(c)}{P_t(c)}\right) P_{t'}(c) \qquad (4)$$

$$cos(C_{t'}, C_t) = \frac{C_t \cdot C_{t'}}{\|C_t\| \, \|C_{t'}\|} \qquad (5)$$

Using two measures focused on different aspects of the distributional similarity between tasks: KL computes the information gain and is asymmetric, cosine similarity computes the normalized dot product between the distributions (as vectors) and is symmetric.

4.2 Mining Similar Tasks

Given that we now have a variety of methods for computing the similarity between pairs of search tasks, the next objective is using that information to mine similar tasks and generate ranking features. In this section we describe the procedure that we employ to leverage task similarity in re-ranking the top retrieved results, as well as the different groups from which we can find similar tasks.

4.2.1 Procedure

Recall that the objective of our method is to find other users attempting the same or similar tasks to the current user. During feature generation, we build a representation of the current search task t based on the queries and result clicks of the user in the task so far, including the current query q (but not its clicks). Each task is modeled using the sources described earlier in Section 3.2.2.

Given that we have constructed a model of t, the next objective is to find similar tasks from the search histories of other users. For each of the tasks t' in the set of all tasks observed historically from other users T, we can then compute the similarity between the current task and those tasks $k(t, t')$. For each of the similarity measures in Section 4.1 we can compute a score s_k for a URL u appearing in the top 10 search results for q:

$$s_k(t, u) = \sum_{t' \in T} \left(k(t, t') \cdot w(t', u)\right) \qquad (6)$$

Where $w(t', u)$ is a weight reflecting the importance of the URL in a related task. In our case we define $w(t', u)$ as the click frequency on that URL for the related task. Other ways of generating this weight are possible (e.g., the rank position of u in the result list), but we focus on click frequency given its computational simplicity and direct relationship with our goal of learning which URLs are relevant from historic on-task behavior. Although T is primarily composed of all historic tasks from all users, it can come from different groups, including the user's long-term history (for personalization rather than groupization), or specific user cohorts such as those with high levels of expertise in the domain of interest.

Once we have s_k for each of the similarity features described in the previous section, we use those values as additional ranking features for each of the results in the top-10 results for q. We then learn to re-rank those results to generate a new result ordering, which we then evaluate based on user behavior as described later.

4.2.2 Re-ranking Features

The specific features that we use for ranking are listed in Table 1. The functions map to the similarity functions described in this section. The inter-task similarity is computed using $k(t, t')$ which is then multiplied against the click count for each URL in the top-10 (i.e., $w(t', u)$ from Equation 6). The result summed over all tasks in the historic data is used to generate the final feature value.

In addition, we also compute *ClickedTasksCount*, which is the total number of tasks for which a particular URL u is clicked. This measures URL popularity independent of task. Note that since *QueryTranslation* and *CategorySimilarityKL* are asymmetric, we also include reverse variants of these features in our feature set.

4.2.3 Groups

The approach we describe in the paper leverages the on-task behavior of groups comprising the following three sets of users:

- **Individual:** This group comprises only the search behavior of the current user. In this group, the queries and similar search tasks are mined only from the current user's long-term history.
- **Group (Global):** In this group, queries and similar tasks are mined from everyone's search histories.
- **Group (Cohorts):** This group comprises particular subsets of *Global* created based on location, browser and search entry point (i.e., how users reach the engine – more details later),

Table 1. Features used to compute the similarity between two tasks, t and t', in the computation of $k(t, t')$.

Feature name	Definition
FullQueryOverlap	The fraction of all queries in the union of t and t' that the two tasks have in common.
QueryTermOverlap	The fraction of all unique query terms in the union of t and t' that the two tasks have in common.
QueryTranslation	Semantic similarity between the queries in t and the queries in t’ ($P(E\|Q, \theta)$ as defined earlier).
ClickedURLOverlap	The fraction of clicked URLs in the union of t and t' that the two tasks have in common.
ClickedDomainOverlap	The fraction of clicked domains in the union of t and t' that the two tasks have in common.
CategorySimilarityKL	The Kullback-Liebler divergence between the ODP category distribution from result clicks in t versus the same distribution from t'.
CategorySimilarityCosine	The cosine similarity between the ODP category distribution from result clicks in t versus the same distribution from t'.

and estimates of topic expertise. The first two are based on information that is readily available in the logs that we used for this study. The latter could be estimated based on patterns of activity, interest on a topic, and success within a topic over time. Queries and similar search tasks in this case are only drawn from the particular cohort (e.g., only from users in the same location as the current searcher) rather than all searchers.

4.2.3.1 Local Cohort

It has been shown that interests can be location specific, e.g., a user querying for [msg] in New York City, NY may be more likely to mean Madison Square Garden than monosodium glutamate [3], and local experts may have better knowledge about the places to select [47]. In this case, we learn our features from users querying from the same location. To identify the user location we use the user’s IP address to determine the city and state for every user. We could not use each {city, state} pair as its own cohort because the population of many locations is insufficient. To address this, we use the city for the most populated locations and back-off to state for the less populated ones. Specifically, the location of a given user is his city if they are in one of the largest 200 U.S. cities by population. Otherwise, the location is the state. For example, a user querying from Austin, Texas would be in the “Austin” cohort, whereas a user querying from College Station, TX would be in the “Texas” cohort.

4.2.3.2 Web Browser / Search Entry Point Cohort

We also created groups of users based on the combination of the Web browser(s) that they use (e.g., Internet Explorer, Firefox, Chrome, multiple browsers, etc.) and the entry point(s) that they use to reach Bing (e.g., Bing homepage, MSN.com, browser search box, multiple entry points, etc.). The determination for each user is based on a held out set of log data from before the time period examined for this paper. Our hypothesis was that users using the same Web browser and entry point may have similar search preferences or be similar demographically (as a recent report by ComScore suggests[1]), and demographics can influence search behavior [44].

4.2.3.3 Topic Cohort

Previous work has shown that people with topic knowledge are more efficient and effective in completing their search tasks [50]. We hypothesized that by focusing on the behavior of experts, we could help users target better quality content. As such, in defining the cohorts we limit the tasks to those from users with significant expertise in the topic of interest. This allows us to learn from expert users in particular, versus learning from one’s personal history or the set of all users, comprising users of all domain expertise levels.

To identify users with significant expertise in different topics, we had to assign topic labels to different queries. We use one of 25 topics to describe any given query. For each such topic, a set of manually-labeled queries are collected from trained judges and a proprietary text classifier is trained using the labeled data. The classifier is then used to assign topics to other queries. We used a set of binary classifiers, one for each topic, allowing queries to belong to multiple topics. Example topics include: *Entertainment*, *Names*, *Commerce*, *Navigational*, *Travel, Technology* and *Sports*.

We use first week of data described in Section 3.1, corresponding to the feature-generation week. A user *U* is deemed to be an expert in topic *P* if the following three conditions are satisfied:

1. **Activity:** The number of queries submitted by *U* is more than the average number of queries per user.
2. **Topic Interest:** The percentage of queries ϵ *P* submitted by *U* exceeds the average percentage of queries ϵ *P* across all users.
3. **Success:** The task success rate of *U* on tasks ϵ *P* is greater than the average task success rate of all users on tasks ϵ *P*. Task success is predicted using the method in [18].

Unlike the location and entry-point cohorts, the expertise cohort does not use information about the current user. The intuition here is that experts will select better resources and being pointed to those resources will help all users irrespective of expertise level. Later we show that there is benefit from leveraging particular user cohorts.

4.3 Summary

In this section we have defined methods for computing inter-task similarity, defined the feature generation procedure and the particular features that are assigned to the URLs, and defined the groups from which similar tasks are drawn. We also described each of the cohorts that we investigate. In the next section we describe our experiments to measure the effectiveness of task-based models for personalization, including comparisons with personalization methods and query-based (not task-based) similarity.

5. EXPERIMENTS

Our log-based evaluation method focuses on a re-ranking task, assessing the extent to which the models promote clicked results.

5.1 Baselines

The original ranking from the Bing search engine is our primary baseline. We also setup competitive query-centric baselines:

1. **Query-based Global (QG; same query, all users):** This is a non-personalized approach that finds clicked URLs by matching the current query against previous queries over all users.

[1] http://www-cs-faculty.stanford.edu/~eroberts/cs181/projects/firefox-market-dynamics/present.html

2. **Query-based Individual (QI; same query, same user):** This finds clicked URLs by matching the current query with previous queries in the current user's search history. This means that if the current query is observed at some point in the user's search history, then the URLs of interest to them then are likely to be promoted in the re-ranked list now. This is similar to the personalized navigation method in Teevan et al. [42].
3. **Query-based Global and Individual Features (QGI):** This is a strong baseline model combining both QG and QI features.

These are very competitive baselines given that they start with the ranking provided by Bing (which already uses behavior data aggregated at the query level) and then add other signals based on the specific query. We focus on the impact of extracting relevance features in similar tasks, rather than exactly matched queries only.

5.2 Research Questions

We utilize two forms of matching, *query-based* and *task-based*. Query-based matches against historic behavioral data based on the exact (normalized) query string of the current query. Task-based matches against historic behavior using the task models and task similarity functions described earlier in the paper.

To evaluate the benefit of task modeling over query modeling, we trained the following nine models using different features, and then compared their performance on the same test data that comprised over two million queries. The nine models evaluated were the three baselines (Models 1-3) plus the following models:

4. **Task-based Global Features (TG):** Trained with features extracted from all tasks in all search histories;
5. **Task-based Individual Features (TI):** Trained with features from tasks in the individual user's search history only;
6. **Task-based Global and Individual Features (TGI):** This model is trained on both TI and TG features;
7. **Query-based and Task-based Global Features (QTG):** This model is trained on both QG and TG features;
8. **Query-based and Task-based Individual Features (QTI):** This model is trained on both QI and TI features;
9. **QTG and QTI Features (QTGI):** This model is trained on both QTG and QTI features.

The re-ranking models attempt to promote observed *satisfied result clicks* (SAT clicks) toward higher rank positions in the result list. This enables offline evaluation of models performance using judgments personalized to each user. This approach has been used to determine the effectiveness of various re-ranking methods [3,5,37].

As described earlier, we attempt to answer the following research questions with our study (we also include the model comparisons):

RQ1: Does matching based on task models outperform matching using the current query? (Models 1-3 vs. Models 4-6).

RQ2: Does task-based groupization outperform task-based personalization? (Model 4 vs. Model 5 vs. Model 6).

RQ3: Is in-session task segmentation required to attain performance gains or would an estimation of tasks as search sessions suffice? (Models 6 and 9 vs. session-based variants)

RQ4: What is the effect of using specific user cohorts for groupization (e.g., those in a particular location or those with good topic knowledge)? [Model 3 vs. (Model 3 + cohorts)].

Answers to these questions help quantify the potential benefits that they can bring to search engine users. Models 7 and 8 are not assigned to any research questions, but are included for completeness.

5.3 Relevance Judgments

Since we were evaluating personalization methods, we needed a personalized relevance judgment for each result. Obtaining many explicit relevance judgments from real users is impractical, and there is no known approach to train expert judges to provide reliable judgments that reflect real user preferences. Hence we obtained these judgments using a log-based methodology inspired by [17] and similar to that used in [3,5,37]. This method infers relevance judgments for query-URL pairs from search-result clicks. We consider three types of clicks in labeling user feedback in the logs: SAT clicks, *quickback* clicks, and no clicks. We define a SAT click in a similar way to previous work [14] as either a click followed by no further clicks for 30 seconds or more, or the last result click in the session. In [14] the authors captured in-situ judgments of satisfaction directly from searchers. This allowed them to determine that a 30-second dwell time was effective in distinguishing satisfaction from dissatisfaction via search behavior alone.

Table 2. Statistics of the weekly data for learning/evaluation.

Count	Training	Validation	Test
SAT Clicks	2,086,335	2,062,554	2,082,145
Quickback Clicks	417,432	408,196	413,496
Tasks	1,165,083	1,126,452	1,135,320
Queries per Task	1.678	1.676	1.666

We define the clicks having less than 30 seconds dwelling time as *quickbacks*. We assign one of the three rating labels to each query-URL pair in the top-10. In each impression, if a URL received at least one SAT click, the URL is labeled with a 2; if a URL received only quickback clicks, the URL is labeled with a 1; if a URL was not clicked at all, the URL is labeled with a 0. This gives us a three-level judgment for each top-10 URL for each query. This multi-level labeling allows the ranker to learn more nuanced differences between the results for each query than could be learned with binary labels. In particular, it helps differentiate between cases where the user explored the page but decided that it was not relevant and cases where they did not consider the URL at all. Since our evaluation methodology is personalized to each user, the relevance labels in *impressions* (unique instances) under the same query could be different, since the users who issue the query vary.

5.4 Measures

We measure ranking quality by mean average precision and mean reciprocal rank. In both cases, the mean is calculated over all the impressions in our test set. Mean average precision (MAP) for a set of queries is the mean of the average precision scores for each query. The average precision score is defined as

$$AveragePrecision = \frac{\sum_{k=1}^{n} Precision(k)Rel(k)}{\sum_{k=1}^{n} Rel(k)} \qquad (7)$$

where n is the number of URLs in the impression, usually 10, $Rel(k)$ is an indicator function equaling 1 if the URL at rank k is a relevant document, zero otherwise, and $Precision(k)$ is the precision at cut-off k in the ranked list.

Mean reciprocal rank (MRR) for a query set is the average of the reciprocal ranks across all results, which is defined as

$$MRR = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{rank_i} \qquad (8)$$

where $rank_i$ is the rank of the first relevant URL in the ranking list, and N is the number of impressions in test.

These measures are complementary in that MRR focuses on the rank of the first relevant document in the top 10, whereas MAP targets the rank of relevant results across the top 10 documents. As

Table 3. MAP/MRR gains on the test data (± SEM). Production ranker is baseline. Query-based baselines highlighted.

Model	Δ MAP(10^{-2})	Δ MRR(10^{-2})	Rerank@1	Coverage	Win	Loss	Cost Rate
QG	**0.0888**±0.0023	**0.1076**±0.0024	0.46%	19.10%	28009	27507	98.21%
QI	**0.1425**±0.0028	**0.1431**±0.0029	0.70%	17.87%	26966	23214	86.09%
QGI	**0.1448**±0.0028	**0.1455**±0.0029	0.71%	19.10%	29259	25097	85.78%
TG	**0.1408**±0.0029	**0.1440**±0.0029	0.88%	67.37%	45866	37668	82.13%
TI	**0.1485**±0.0028	**0.1490**±0.0029	0.71%	19.44%	30932	26586	85.95%
QTI	**0.1691**±0.0030	**0.1695**±0.0030	0.79%	20.23%	30193	25180	83.40%
QTG	**0.1905**±0.0032	**0.1936**±0.0032	1.01%	67.55%	33102	23617	71.35%
TGI	**0.2292**±0.0035	**0.2318**±0.0036	1.22%	67.37%	32753	22292	68.06%
QTGI	**0.2516**±0.0036	**0.2542**±0.0037	1.28%	67.55%	35425	24731	69.81%

such, MAP can also measure performance in queries with multiple clicks. In test, only the URLs that received SAT clicks are considered as relevant, and the URLs received only quickback clicks are not treated as relevant URLs in evaluation. During testing, we wanted to be conservative and only regard results as relevant for which we could be most confident that searchers were satisfied.

In addition to measuring the relevance of the results, we also measure the coverage of the models in two main ways: the fraction of the results at the top-position (rank=1) that are re-ranked by the approach and the fraction of all impressions covered by relevance features. The re-ranking coverage at the most prominent position indicates the extent of the time where the re-ranking signal from the model is extremely strong. Feature coverage indicates fraction of impressions for which a signal is available (e.g., a similar task can be found in all users' search histories).

5.5 Method

We use the four weeks of logs described in Section 3.1 for our experiments. For all of the methods under test, we used the first week of logs for feature generation (i.e., computing the scores (s_k) for the clicked URLs in the first week of data as described earlier in the paper), the second week for model training, the third week for model validation, and the fourth week for testing. Logs were collected from A/B tests where other personalization support was disabled, so as to not bias our results with other personalization signals. Table 2 presents summary statistics on the three data sets used. Each set contains around 2 million impressions, which is less than the 3 million reported earlier since we drop impressions without any SAT clicks. We evaluated the significance of observed differences across all queries in the test week using paired t-tests with the significance level (α) set to α=0.05. When performing multiple comparisons, Bonferroni corrections are performed to reduce the likelihood of Type I errors (i.e., incorrectly rejecting a true null hypothesis) by dividing α by the number of pairs under comparison.

Using the described dataset, we trained a ranking model using the LambdaMART learning algorithm [48] for re-ranking the top ten search results. LambdaMART is an extension of LambdaRank [9] based on boosted decision trees. LambdaMART has been shown to be one of the best algorithms for learning to rank. Indeed, an ensemble model in which LambdaMART rankers were the key component won Track 1 of the 2010 Yahoo! *Learning to Rank Challenge* [10]. However, we note the choice of learning algorithm is not central to this work, and any reasonable learning to rank algorithm would likely provide similar results.

6. RESULTS

We now present the results of our analysis, broken out by each of the four research questions described in Section 5.2. Results are primarily reported as averages over the 2 million test queries. Recall that the goal is to re-rank the results *for a given query* using the on-task behavior of the current searcher or other searchers (everyone or particular cohorts of similar users). By varying the model comparisons as suggested in Section 5.2, we can answer each of our research questions. Where appropriate, we analyze the effect of various properties on the results, in particular the number of tokens in the query and the relative position of the query in the search task, supporting the construction of rich interest models. To provide a good sense of the overall impact of the models, we report re-ranking performance across the full set of queries, including many of the queries that have no change in the ranking. Including these queries drove the mean average change in MAP and MRR toward zero, but more fully reflected the overall effect of the models than, say, focusing on the average over queries where metrics changed.

6.1 Task Matching vs. Query Matching

Table 3 reports the MAP/MRR gains of each model versus the baseline (production ranker used in Bing at the time the log data was captured) ± the standard error of the mean (SEM). All differences with that baseline are significant at p < 2.2e-16. As described earlier, the effect of task modeling is measured by comparing the three query-based models (QG, QI, and QGI) with the three task-based variants that use the same sources (TG, TI, and TGI). To test the significance of the observed differences, between models we performed a two-way analysis of variance (ANOVA) with matching method and group as factors. We also computed the effect size of the observed differences using *partial eta squared* (η_p^2), a commonly used measure of effect size in analyses of variance. The main effects of matching method and group were significantly different at $p < 0.001$ ($F_{\text{Matching}}(1,12492864) = 12.94$, $F_{\text{Group}}(2,12492864) = 6.76$). In addition, the matching-group interactions were significant ($F_{\text{Matching}\times\text{Group}}(2,12492864) = 4.82$, $p < 0.01$; Tukey post-hoc testing: all $p < .001$). This was expected given the large sample sizes, but the interaction effect was small in magnitude (i.e., $\eta_p^2 = 0.02$).

Table 3 also presents the feature coverage of each of the nine models, and the win and loss counts in test. The win and loss were determined by the MAP metric. If the re-ranked order results in a positive MAP gain over the baseline model, we count it as a win; it is counted as a loss if the re-ranked order yields negative MAP change against the control model. This helps us to understand the trade-offs between risk and reward in the different methods (i.e., given equal average MAP gains between two models, we would prefer the model with the lower cost rate). Global features yield the largest feature coverage. QTGI (that combines all signals) performs best overall in terms of both MAP and MRR, and the TGI model (combining personalization- and groupization-based task modeling) has the best re-ranking performance in terms of cost rate.

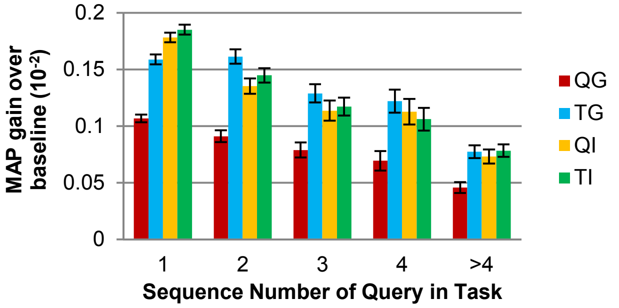

Figure 2. Segment analysis on MAP for queries issued at different points in the task (± SEM).

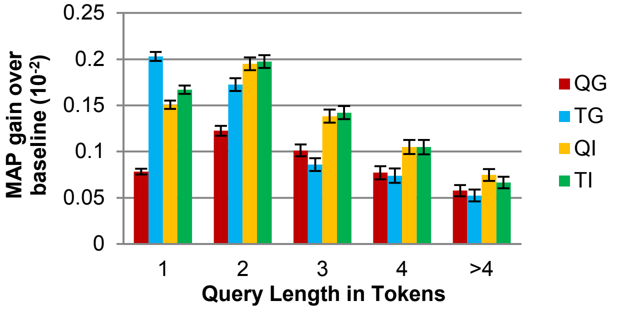

Figure 3. Segment analysis on MAP performance for queries of different lengths (± SEM).

In the next experiment we broke out the experimental results by two conditions, current query length and sequence number of the query in the task. Figure 2 shows that the performance of the two key group variants (group and individual) and the two matching variants (query and task). The figure shows that all models consistently outperform the QG at all points in the search task. The comparison between TG and QG is particularly relevant in this section because it demonstrates the benefit of task-based matching over query-based matching. The three other models are comparable across the range of query positions. The relative drop in MAP over the course of the session is of a similar extent across all queries. The decrease in gain from personalization approaches has been observed in other personalization research [5]; possible explanations include queries becoming more specific as the session proceeds, making the retrieval task more difficult and reducing the potential benefit from personalization, longer tasks being more difficult generally, or that the first query in the session found most of the relevant information.

Figure 3 reports the performance of each of the models as the length of the current query varies. The chart shows once again that TI and QI are comparable across the range of query lengths. TG outperforms QG for queries of length one or two (and the personalized methods QI and TI for queries of length one), and has comparable performance for longer queries (with individual methods performing slightly better). One explanation for this is that for shorter queries, searchers provide specific information that they have searched for before (in which case QI and TI do well) or only providing partial information meaning that identifying resources accessed by others attempting similar tasks might be useful (TG does well).

6.2 Group vs. Individual

An important consideration is the value of the task-based groupization compared to task-based personalization. The next question we considered was the differences in performance between these two methods (i.e., TG vs. TI). We compare re-ranking results using task-based matching against the current user's history (TI) with those of using all history of all users (TG). We also considered how well the methods perform in combination (TGI). Table 4 summarizes the findings, which show that task-based groupization and task-based personalization perform similarly even though the personalized variant is tailored to the current user (TI vs. TG, both $t(2082143) \leq 1.84$, both $p \geq 0.0656$, α = 0.025). Therefore, the group-based ranking signal may be a sufficient approximation for personalization and as we show in Table 3, and it has the big advantage of covering many more queries (67% vs. 19%). Interestingly, we also note that the two features cooperate well in the combined model, TGI, leading to significant gains over the baseline.

Table 4. Comparison on the test data. ΔMAP and ΔMRR denote the MAP and MRR difference from the baseline model (TG) respectively (± SEM).

Models	ΔMAP(10^{-2})	ΔMRR(10^{-2})
TI vs. TG	**0.0077**±0.0033	**0.0050**±0.0025
TGI vs. TG	**0.0884**±0.0026	**0.0878**±0.0031

Table 5. Test results on the test data. ΔMAP and ΔMRR denote the MAP and MRR difference from the baseline model (the original ranking) respectively (± SEM).

Models	ΔMAP(10^{-2})	ΔMRR(10^{-2})	Rerank@1
SGI	**0.2134**±0.0034	**0.2170**±0.0035	1.22%
QSGI	**0.2497**±0.0037	**0.2526**±0.0037	1.28%

Table 6. Comparison on the test data. ΔMAP and ΔMRR denote the MAP and MRR difference from the baseline models (SGI and QSGI) respectively (± SEM).

Models	ΔMAP(10^{-2})	ΔMRR(10^{-2})
TGI vs. SGI	**0.0158**±0.0022	**0.0147**±0.0022
QTGI vs. QSGI	**0.0019**±0.0021	**0.0016**±0.0022

6.3 Task vs. Session

Another important consideration is the value of segmenting the tasks into sessions, versus simply using the full session. If we could attain similar performance to tasks using temporally-delimited sessions then, for computational simplicity and reduced overhead, we may want to simply use sessions as a proxy for task. To evaluate the benefit of task-based modeling over session modeling, we used the following two comparator models replacing tasks with sessions:

1. **Session-based Global and Individual Features (SGI):** This model is trained on both SI and GI features, which is compared against the TGI model on the same test samples, and;
2. **QGI and SGI Features (QSGI):** This model is trained on both QGI and SGI features, which is compared against the QTGI model on the same test samples.

Table 6 reports the MAP and MRR differences between the TGI model and SGI, directly comparing tasks with sessions. The results of that comparison show that the task-based approach significantly outperforms the session-based method (both $t(2082143) \geq 6.76$, both $p < 1.4\text{e-}11$, α = 0.025). When we also consider the query-based matching features as part of the comparison (QTGI vs. QSGI) gain observed from the task representation becomes non-significant (both $t(2082143) \leq 1.31$, both $p > 0.1905$, α = 0.025). This suggests that much of the gain from the task modeling over the session modeling comes from being able to match based on the same or similar queries, which seems reasonable given that the task modeling generates focused query clusters by design. When query is factored into the model directly (as is the case in moving from TGI to QTGI) then the benefit from using tasks over sessions diminishes.

Table 7. MAP/MRR gains for cohorts on the test data (± SEM). QGI is baseline.

Models	Δ MAP(10^{-2})	Δ MRR(10^{-2})	Rerank@1	Coverage	Win	Loss	Cost Rate
QGI+Local	**0.0505**±0.0019	**0.0508**±0.0019	0.38%	39.94%	12777	9965	77.99%
QGI+Topic	**0.0851**±0.0024	**0.0872**±0.0024	0.66%	28.40%	38973	33523	86.02%
QGI+Entry	**0.0646**±0.0021	**0.0661**±0.0021	0.48%	23.08%	23018	19453	84.51%

6.4 Effect of User Cohorts

In addition to using the on-task behavior of all users, we also studied the effect of using the three cohorts described earlier in the paper. Our hypothesis was that using the behavior of users who are similar to the current user or who are in some way knowledgeable about the topic of interest would boost retrieval performance. Table 7 summarizes the results when compared against the QGI baseline model. This was our strongest baseline and allowed us to examine the effect of cohorts without conflating task with cohort. All differences with baseline are significant at p < 2.2e-16.

The results show clearly that cohorts improve performance over the strong baseline. To directly compare the effect of the different cohorts we used a one-way ANOVA. The results of this analysis show that the differences between the models were significant (F(2, 6246432) = 15.13, p < 0.001; Tukey post-hoc test: all p < 0.001). The best performing cohort model (on average over all queries) was the topic expertise model, suggesting that users may benefit from focusing on the web sites that experts visit. However, the local-cohort model is less risky (lower cost rate) and also covers a larger fraction of the queries. More work is needed to understand the cost-benefit tradeoffs of using cohorts, as well as how cohorts interact.

6.5 Summary

The results of our study show that:

1. Task-based matching to historic data outperforms query-based matching, both in terms of relevance and coverage (RQ1).
2. Task-based groupization has statistically indistinguishable performance from task-based personalization, but has dramatically better coverage (over 3 times greater) (RQ2).
3. Task-based segmentation methods lead to gains in performance over sessions, suggesting that there is value in first grouping session activity into coherent task clusters (RQ3).
4. Leveraging the behavior of particular cohorts rather than all users leads to better performance than a strong baseline (RQ4).

7. DISCUSSION AND IMPLICATIONS

We have presented a study on mining and modeling search tasks to improve search personalization. Our novel approach mines similar tasks from other users and uses them for re-ranking, improving coverage while attaining similar performance gains to traditional personalization methods. However, more detailed analysis of the findings is required to understand exactly when the personalization and groupization approaches are most successful and when to choose between them. Further improvements in performance may well be observed given a broader range of task-oriented features than the modest set employed in this paper. Our main contribution is as the first study to show performance gains via groupization by implicitly modeling all users', and user cohorts', on-task behavior at scale.

Although we showed promising gains with cohort modeling over a strong query-based method, more work is needed to understand how task models can be enhanced using cohort information. Early experiments with TGI + cohort revealed no significant gains over TGI, and it might be the case that cohorts only help when needs are specific. More sophisticated cohort modeling could be employed to leverage other information such a social relationships, available via social networks (e.g., focus on the on-task behavior of friends), or those with similar interests to the current user outside of the current query topic (and hence likely to have similar preferences). The topic cohort focused on finding experts, irrespective of the current user's expertise; modeling *relative* expertise may help.

The success of our approach is dependent on how accurately we can model search tasks. The approach described in this paper was useful to demonstrate the potential value of this method, but more sophisticated models of search tasks could be developed to include signals such as task success [18], e.g., so that we focus on tasks where the outcome was successful or demote unsuccessful tasks. We also focused on behaviors on the search engine (queries and result clicks). However, there may be valuable information in considering search behavior once users click a result and navigate away from the engine [46] or those sites that users target directly without using a search engine, especially for users with domain expertise. The challenge in the latter case is generalizing task modeling to extend beyond search activity and allow a mapping between search tasks and these more general task representations. Richer models could also be developed by considering search and usage behaviors which may not be logged at server side (e.g., document retention events such as printing and bookmarking). In addition, although the current approach focused on re-ranking (mainly necessitated by the need for personalized relevance judgments), the best results may not be available at top positions, and deeper re-ranking or even the injection of non-indexed URLs into the list needs to be considered.

8. CONCLUSIONS

We have studied methods for modeling users' on-task search behavior and using those models to improve personalization methods. We focused on a scenario where by building rich models of the current user's task we can find other users who have performed similar tasks historically, and leverage their on-task behavior to improve personalization performance. We show though extensive experimentation that our methods outperform query-based personalization methods that use the current user's long-term search history, as well as other approaches that match with aggregated behavior of many searchers based on the text of the search query. This clearly demonstrates the value of considering search *tasks* rather than just search *queries* during personalization, as well as the benefit of groupization. Mining on-task behavior from particular cohorts (rather than all users) was also shown to be useful, at least for query-based matching. Future work involves the use of a broader range of cohorts and cohort combinations, and the development of more sophisticated and generalizable models of task behavior that can mine and model task-relevant activity beyond search engine interactions.

REFERENCES

1. Agichtein, E., Brill, E., and Dumais, S. (2006). Improving web search ranking by incorporating user behavior information. *SIGIR*, 19–26.
2. Almeida, R. and Almeida, V. (2004). A community-aware search engine. *WWW*, 413–421.
3. Bennett, P.N., Radlinski, F., White, R.W., and Yilmaz, E. (2011). Inferring and using location metadata to personalize web search. *SIGIR*, 135–144.

4. Bennett, P., Svore, K., and Dumais, S. (2010). Classification-enhanced ranking. *WWW*, 111–120.
5. Bennett, P., White, R.W., Chu, W., Dumais, S., Bailey, P., Borisyuk, F., and Cui, X. (2012). Modeling the impact of short and long-term behavior on search personalization. *SIGIR*, 185–194.
6. Berger, A.L. and Lafferty, J. (1999). Information retrieval as statistical translation. *SIGIR*, 222–229.
7. Bilenko, M. and White, R.W. (2008). Mining the search trails of surfing crowds: identifying relevant websites from user activity. *WWW*, 51–60.
8. Brown, P.F., Della Pietra, S.A., Della Pietra, V.J., and Mercer, R.L. (1993). The mathematics of statistical machine translation: parameter estimation. *Comp. Ling.*, 19(2): 263–311.
9. Burges, C.J.C., Ragno, R., and Le, Q.V. (2006). Learning to rank with non-smooth cost functions. *NIPS*, 193–200.
10. Chapelle, O., Chang, Y., and Liu, T.-Y. (2010). Yahoo! learning to rank challenge. *learningtorankchallenge.yahoo.com.*
11. Dou, Z., Song, R., and Wen, J.R. (2007). A large-scale evaluation and analysis of personalized search strategies. *WWW*, 581–590.
12. Downey, D., Dumais, S.T., Liebling, D., and Horvitz, E. (2008). Understanding the relationship between searchers' queries and information goals. *CIKM,* 449–458.
13. Freyne, J. and Smyth, B. (2006). Cooperating search communities. *AH*, 101–110.
14. Fox, S., Karnawat, K., Mydland, M., Dumais, S.T., and White, T. (2005). Evaluating implicit measures to improve the search experience. *TOIS*, 23(2): 147–168.
15. Gao, J., He, X., and Nie, J.-Y. (2010). Clickthrough-based translation models for web search: from word models to phrase models. *CIKM*, 1139–1148.
16. Gao, J., Xie, S., He, X., and Ali, A. (2012). Learning lexicon models from search logs for query expansion. *EMNLP*.
17. Gao, J., Yuan, W., Li, X., Deng, K., and Nie, J.-Y. (2009). Smoothing clickthrough data for web search ranking. *SIGIR*, 355–362.
18. Hassan, A. (2012). A semi-supervised approach to modeling web search satisfaction. *SIGIR*, 275–284.
19. Hassan, A., Jones, R., and Klinkner, K. (2010) Beyond DCG: user behavior as a predictor of a successful search. *WSDM*, 221–230.
20. Hassan, A., Song, Y., and He, L. (2011). A task-level metric for measuring web search satisfaction and its application on improving relevance estimation. *CIKM,* 125–134.
21. Joachims, T. (2002). Optimizing search engines using clickthrough data. *KDD*, 133–142.
22. Jones, R. and Klinkner, K.L. (2008). Beyond the session timeout: automatic hierarchical segmentation of search topics in query logs. *CIKM*, 699–708.
23. Kellar, M., Watters, C., and Shepherd, M. (2007). A field study characterizing Web-based information-seeking tasks. *JASIST*, 58(7): 999–1018.
24. Kotov, A., Bennett, P.N., White, R.W., Dumais, S.T., and Teevan, J. (2011). Modeling and analysis of cross-session search tasks. *SIGIR*, 5–14.
25. Lee, Y-J. (2005). VizSearch: a collaborative web searching environment. *Computers and Education*, 44(4): 423–439.
26. Li, Y. and Belkin, N.J. (2008). A faceted approach to conceptualizing tasks in information seeking. *IP&M*, 44(6): 1822–1837.
27. Liao, Z., Song, Y., He, L.W., and Huang, Y. (2012). Evaluating the effectiveness of search task trails. *WWW*, 489–498.
28. Liu, J. and Belkin, N.J. (2010). Personalizing information retrieval for multi-session tasks: the roles of task stage and task type. *SIGIR*, 26–33.
29. Liu, J., Cole, M.J., Liu, C., Bierig, R., Gwizdka, J., Belkin, N.J., Zhang, J., and Zhang, X. (2010). Search behavior in different task types. *JCDL*, 69–78.
30. Lucchese, C., Orlando, S., Perego, R., Silvestri, F., and Tolomei, G. (2011). Identifying task-based sessions in search engine query logs. *WSDM*, 277–286.
31. MacKay, B. and Watters, C. (2008). Exploring multi-session web tasks. *SIGCHI*, 1187–1196.
32. Mei, Q. and Church, K. (2008). Entropy of search logs: how hard is search? with personalization? with backoff? *WSDM*, 45–54.
33. Radlinski, F. and Joachims, T. (2005). Query chains: learning to rank from implicit feedback. *KDD*, 239–248.
34. Radlinski, F., Szummer, M. and Craswell, N. (2010). Inferring query intent from reformulations and clicks. *WWW*, 1171–72.
35. Shen, X., Tan, B., and Zhai, C. X. (2005). Implicit user modeling for personalized search. *CIKM*, 824–831.
36. Smyth, B. (2007). A community-based approach to personalizing Web search. *IEEE Computer*, 40(8): 42–50.
37. Sontag, D., Collins-Thompson, K., Bennett, P.N., White, R.W., Dumais, S.T., and Billerbeck, B. (2012). Probabilistic models for personalizing web search. *WSDM*, 433–442.
38. Sugiyama, K., Hatano, K., and Yoshikawa, M. (2004). Adaptive Web search based on user profile constructed without any effort from users. *WWW*, 675–684.
39. Sun, J.-T., Zeng, H.-J., Liu, H., Lu, Y., and Chen, Z. (2005). CubeSVD: a novel approach to personalized web search. *WWW*, 382–390.
40. Tan, B., Shen, X. and Zhai, C. (2006). Mining long-term search history to improve search accuracy. *KDD*, 718–723.
41. Teevan, J., Dumais, S. T., and Horvitz, E. (2005). Personalizing search via automated analysis of interests and activities. *SIGIR*, 449–456.
42. Teevan, J., Liebling, D.J., and Geetha, G.R. (2011). Understanding and predicting personal navigation. *WSDM*, 85–94.
43. Teevan, J., Morris, M.R., and Bush, S. (2009). Discovering and using groups to improve personalized search. *WSDM*, 15–24.
44. Weber, I. and Castillo, C. (2010). The demographics of web search. *SIGIR*, 523–530.
45. White, R.W., Bailey, P. and Chen, L. (2009). Predicting user interests from contextual information. *SIGIR*, 363–370.
46. White, R.W., Bennett, P.N., and Dumais, S.T. (2010). Predicting short-term interests using activity-based search context. *CIKM*, 1009–1018.
47. White, R.W. and Buscher, G. (2012). Characterizing local interests and local knowledge. *SIGCHI*, 1607–1610.
48. Wu, Q., Burges, C.J.C., Svore, K.M., and Gao, J. (2008). Ranking, boosting, and model adaptation. *Microsoft Research Technical Report MSR-TR-2008-10.*
49. Xiang, B., Jiang, D., Pei, J., Sun, X., Chen, E., and Li, H. (2010). Context-aware ranking in web search. *SIGIR*. 451–458.
50. Zhang, X., Anghelescu, H.G.B., and Yuan, X. (2005). Domain knowledge, search behavior, and search effectiveness of engineering and science students. *Inf. Res.*, 10(2): 217.