Mining Search and Browse Logs for Web Search: A Survey

Daxin Jiang Jian Pei Hang Li
Microsoft Corporation Simon Fraser University Huawei Technologies
Email: djiang@microsoft.com jpei@cs.sfu.ca hangli.hl@huawei.com

Huge amounts of search log data have been accumulated at web search engines. Currently, a popular web search engine may every day receive billions of queries and collect tera-bytes of records about user search behavior. Beside search log data, huge amounts of browse log data have also been collected through client-side browser plug-ins. Such massive amounts of search and browse log data provide great opportunities for mining the wisdom of crowds and improving web search. At the same time, designing effective and efficient methods to clean, process, and model log data also presents great challenges.

In this survey, we focus on mining search and browse log data for web search. We start with an introduction to search and browse log data and an overview of frequently-used data summarizations in log mining. We then elaborate how log mining applications enhance the five major components of a search engine, namely, query understanding, document understanding, document ranking, user understanding, and monitoring & feedbacks. For each aspect, we survey the major tasks, fundamental principles, and state-of-the-art methods.

Categories and Subject Descriptors: ... [...]: ...

General Terms: ...

Additional Key Words and Phrases: ...

1. INTRODUCTION

Huge amounts of search log data have been accumulated in various search engines. Currently, a commercial search engine receives billions of queries and collects tera-bytes of log data on every single day. Other than search log data, browse logs have also been collected by client-side browser plug-ins, which record user browse information if users' permissions are granted. Such massive amounts of search/browse log data, on the one hand, provide great opportunities to mine the wisdom of crowds and improve web search results. On the other hand, designing effective and efficient methods to clean, model, and process large scale log data also presents great challenges.

The objective of this survey is threefold.

—First, we provide researchers working on search/browse log mining or the related problems a good summary and analysis of the state-of-the-art methods and a stimulating discussion on the core challenges and promising directions. Particularly, for researchers planning to start investigations in this direction, the survey can serve as a short introduction course leading them to the frontier quickly.

—Second, we provide general data mining audience an informative survey. They can get a global picture of the state-of-the-art research on search and browse log data mining. Moreover, researchers in other fields who need to tackle problems in similar nature can quickly understand the on-the-shelf techniques that they

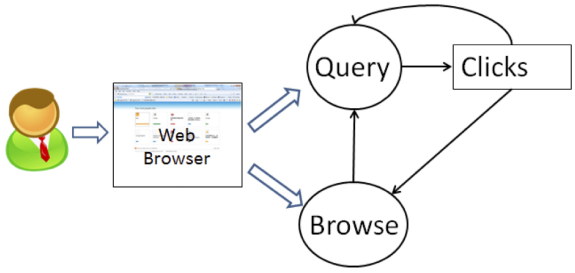

Fig. 1. A user's behavior in a web browser can be categorized into two states: the search state and browse state.

can borrow to solve their problems.

—Third, we provide industrial search engine practitioners a comprehensive and in-depth reference to the advanced log mining techniques. We try to bridge the research frontier and the industrial practice. The ideas and solutions introduced in this survey may motivate the search engine developers to turn research fruits into product reality.

In this section, we first describe the major content recorded in search and browse logs, respectively. Since raw log data is usually noisy and of extremely large size, pre-processing is often conducted before mining algorithms are applied. In the second part of this section, we will introduce four data summarizations, which are created in pre-processing and widely used in log mining. After that, we will present an overview of log mining technologies and applications in web search in the last part of this section.

1.1 Major Content in Logs

To better understand what search and browse logs record, let us first consider how people access information on the web through browsers. Users' behavior in a web browser can be categorized into two states, namely, the *search state* and the *browse state* (see Figure 1).

In the search state, a user raises a query to a search engine and selectively clicks on the search results returned by the search engine. After that, the user may further refine the query and continue the interaction with the search engine. When the user visits a web page other than a search result page, she is considered to be in the browse state. The major difference between the search state and the browse state lies in the type of server which the user interacts with: if a user is interacting with a search engine, she is in the search state; otherwise, she is in the browse state.

Users often make transitions between the search state and the browse state. When a user browses a web page, she may want to go to a search engine and search contents related to the page. In this case, the user transits from the browse state to the search state. On the other way, a user may also transit from the search state to the browse state. To be specific, after the user clicks on a web page in the search result, she may further follow the hyper-links of the web page and leave the interaction with the search engine.

There exist various types of log data, and each targets at some specific user actions and can be collected at either the server side or the client side. In this survey, we focus on two types of log data, namely, *search logs* and *browse logs*.

Search logs are collected by search engines and record almost all interaction details between search engines and users, including queries submitted to search engines, search results returned to users, and clicks made by users. In general, there are four categories of information in search log data: (1) user information, such as IP addresses and machine-generated user IDs, (2) query information, such as text and timestamps of queries, (3) click-through information, such as URLs and timestamps of clicks, as well as positions of clicked URLs, and (4) search results provided by search engines, such as document ranking results and advertisement results.

Browse logs are usually collected by client-side browser plug-ins or proxies of Internet Service Providers. They record all URLs visited by users, no matter from search engines themselves or other web servers. Therefore, we may extract from browse logs not only users' browsing behaviors but also users' search behaviors. To that extent, browse logs provide a more comprehensive picture of user behaviors than search logs do. In addition, browse logs also contain URLs and timestamps of web pages browsed by the users. Browse logs, however, usually do not contain search results returned from search engines. To connect search results and click information, certain data integration processing is necessary.

Importantly, collecting and using browse log data must strictly follow some well defined privacy policy meeting the proper regulation. Browse log data may be collected only under users' permissions. Moreover, users should be able to easily opt out from browse log data collection.

1.2 Frequently Used Data Summarizations

Although search and browse log data provide great opportunities for enhancing web search, there are several challenges before such data can be used in various applications. First, the size of log data is usually very large. In practice, the size of search and browse log data at a search engine often at the magnitude of tens of tera-bytes each day. Second, log data are quite noisy. For example, queries may be issued by machines for experiments; user input in URL boxes may be redirected to search engines by web browsers; and clicks on search result pages may be randomly made by users.

To overcome noise and volume, one can aggregate raw log data in pre-processing. By summarizing common patterns in raw data, the size of data can be greatly reduced. Moreover, after aggregation, we may prune patterns with low frequencies to reduce noise.

One question is how to summarize raw log data for various log mining tasks. In fact, search and browse log data have very complex data structures with various types of data objects and relationships, as illustrated in Figure 2. The data objects may include users, sessions, queries, search result pages, clicks on search results, and follow-up clicks. These different types of data objects form a hierarchy. At the top level, each user has a series of sessions, where each session contains a sequence of queries. In a query, a user may open several web pages. Finally, a user may further follow the hyperlinks in the web pages of search results and browse more

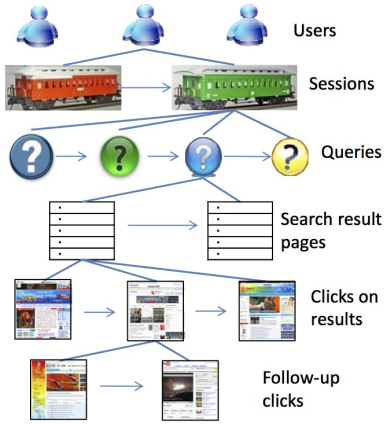

Fig. 2. Although search and browse log data have complex data structures, they can be summarized in a hierarchy of data objects.

Query	Count
facebook	3,157 K
google	1,796 K
youtube	1,162 K
myspace	702 K
facebook com	665 K
yahoo	658 K
yahoo mail	486 K
yahoo com	486 K
ebay	486 K
facebook login	445 K

Fig. 3. An example of query histogram, which consists of queries and their frequencies.

web pages. In addition to the hierarchical relationship between different types of data objects, the data objects at the same level often form a sequential relationship.

Here, we introduce four types of data summarization that are widely used in log mining, namely, *query histograms*, *click-through bipartites*, *click patterns*, and *session patterns*. Among the literature reviewed in this survey, 90% of the papers on log mining utilized at least one of the four types of data summarization.

1.2.1 *Query histogram.* A query histogram represents the number of times each query is submitted to a search engine. As shown in Figure 3, query histogram contains query strings and their frequencies. As a simple statistics, query histogram can be used in a wide variety of applications, such as query auto completion and query suggestion.

1.2.2 *Click-through bipartite.* A click-through bipartite graph, such as Figure 4, summarizes click relations between queries and URLs in searches. The bipartite graph consists of a set of query nodes and a set of URL nodes. A query and a URL are connected by an edge if the URL is clicked by a user when it is returned as an

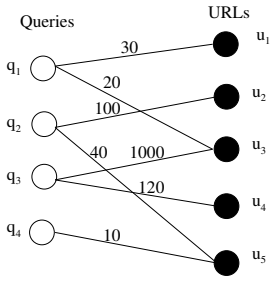

Fig. 4. An example of click-through bipartite graph. In a click-through bipartite graph, nodes represent queries and URLs, and edges represent click relations between queries and URLs.

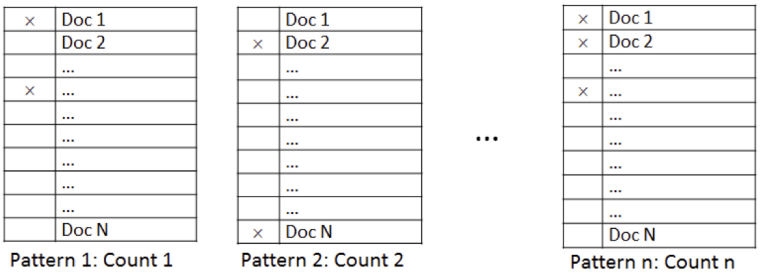

Fig. 5. An illustration of click patterns. Click patterns summarize positions of clicked URLs in search results of queries.

answer to the query. A weight c_{ij} may be associated with an edge e_{ij} indicating the total number of times URL u_j is clicked with respect to query q_i. Click-through bipartite is probably the most widely used data structure in log mining. As we will see in the following sections, it can be used for query transformation, query classification, document annotation, and many other tasks.

1.2.3 *Click patterns.* Click patterns summarize positions of clicked URLs in search results of queries. To be specific, each search result (also known as search impression) $\mathcal{I}_q$ with regard to query q can be represented by $\mathcal{I}_q = (q; L)$, where L is a list of triples (u, p, c), where u is the URL of a page, p is the position of the page, and c indicates whether the page is clicked. The identical search results are further aggregated to one click pattern $\mathcal{P}_q = (q; L; cc)$, where cc is the number of search results. Figure 5 illustrates examples of click patterns. In practice, a list L only includes the top N URLs. Compared with a click-through bipartite, click patterns contain richer information. A click-through bipartite only represents aggregated clicks of URLs, while click patterns further represent the positions of the clicked URLs as well as unclicked URLs. As will be seen in the later sections, click patterns can facilitate many tasks in search, such as classifying navigational and informational queries, learning pairwise document preference, building sequential click models, and predicting user satisfaction.

1.2.4 *Session patterns.* Session patterns summarize transitions among queries, clicks, and browses within search sessions. In fact, session patterns can be defined in different ways depending on specific applications. For example, Cao et al. [2008] and Boldi et al. [2008] take sequences of queries as sessions, and extract frequent query sequences as session patterns. In other cases, session patterns may involve not only queries, but also clicked URLs. For example, Cao et al. [2009] defined session patterns based on sequences of queries and their clicked URLs. Since session patterns represent users' search behaviors in a more precise way, it has been used extensively. As will be seen later, session patterns have been widely used in tasks such as query transformation, document ranking, and user satisfaction prediction.

One critical issue with regard to session patterns is to determine the boundaries of sessions in a query stream from the same user. A widely used simple method is the so-called "30 minute rule". That is, any time interval longer than 30 minutes can be regarded as a boundary [Boldi et al. 2008]. Jones and Klinkner [2008] formalized the problem of session boundary detection as a classification problem. That is, given two adjacent queries in a query stream, decide whether they belong to two sessions or not. Their classifier makes use of features like the length of time between the two queries, the word and character similarities between the two queries, the statistical co-occurrences of the two queries, and the similarities of search results between the two queries. Jones and Klinkner [2008] showed that they can significantly enhance the precision from 70% to 92% using solely temporal features in the classification approach. See also [He et al. 2002; Lucchese et al. 2011] for other approaches to session segmentation.

1.3 Log Mining Technologies and Applications in Web Search

With the abundance of search and browse log data, numerous log mining technologies have been developed. Silvestri [2010] divided log mining technologies into two major categories. The first category focuses on enhancing the efficiency of a search system, while the second category focuses on enhancing the effectiveness of a system. In this survey, we mainly introduce the technologies in the latter category, because most of the technologies are about effectiveness. There are also some other surveys on the topic, such as [Baeza-Yates 2004; Agichtein 2010]. Those surveys mainly focus on query understanding, while this survey covers five aspects of how log mining enhances web search, namely, query understanding, document understanding, document ranking, user understanding, and monitoring & feedbacks.

1.3.1 *Overview of web search system.* From the viewpoint of effectiveness, a web search system usually consists of three basic components, namely, *query understanding*, *document understanding*, and *document ranking*, and one optional component, namely *user understanding*, as shown in Figure 6.

—Document understanding is to transform web pages into some representations that cover both content and importance. This task is usually carried out offline.

—Query understanding is performed online. Specifically, it receives queries and transforms them into some representations.

—Document ranking is conducted to retrieve and rank documents with respect to a query based on the query and document representations. A ranking model

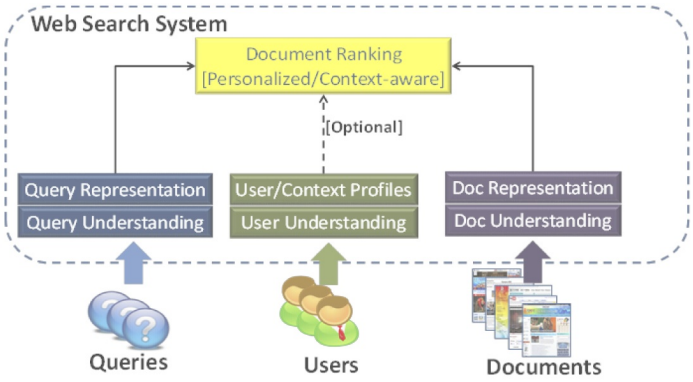

Fig. 6. The architecture of a web search system. A web search system usually consists of query understanding, document understanding, document ranking, and user understanding

typically has the form $P(\mathcal{M}_d|\mathcal{M}_q)$, where $\mathcal{M}_q$ is the query representation of a query q and $\mathcal{M}_d$ is the document representation of a document d.

The above three components constitute a search system. In recent years, the importance of users is emphasized and the component of user understanding is also considered. The task of user understanding is to analyze user behavior and create either user profiles or context profiles in order to rank documents better. In general, a user profile summarizes from a user's long search and browse history and characterizes the user's preferences, while a context profile usually focuses on the environment, such as time, location, and search device, of a specific search activity. User profiles and context profiles will be described in detail in Sections 5.1 and 5.2, respectively. Given a user profile $\mathcal{P}_u$ or a context profile $\mathcal{P}_c$, the ranking model can be represented as $P(\mathcal{M}_d|\mathcal{M}_q, \mathcal{P}_u)$ or $P(\mathcal{M}_d|\mathcal{M}_q, \mathcal{P}_c)$.

In addition to the above components, monitoring a search system and collecting user feedbacks are also important for the effectiveness of a system.

1.3.2 *Enhancing query understanding.* Understanding the search intent of a query has been recognized as a crucial part of effective information retrieval. Despite this, search systems, in general, have not focused on explicitly representing query intents. Query processing has been limited to simple transformations such as stemming or spelling correction.

In web search, with large amounts of search and browse log data available, it becomes possible to conduct deeper query understanding and represent intents of queries with richer representations. Query understanding includes methods for various tasks, including (1) query substitution, (2) query classification, (3) query transformation, (4) query segmentation, and (5) named entity recognition in queries. Section 2 will review the existing work on using search and browse log mining to enhance those tasks.

1.3.3 *Enhancing document understanding.* Web pages are hypertext documents containing HTML tags and links. One can create rich document representations for web pages and leverage them in web search. Actually, the uses of the tagging

and linking information of web pages have been proved to be very effective. The anchor texts of web pages can be utilized as external tags to pages. The HTML formatting information of web pages is useful for identifying titles as well as key phrases of pages. Furthermore, the link graph of the web pages can be used to judge page importance by applying the PageRank or HITS algorithms.

Search and browse log data provide new opportunities to enhance the representations of documents in both content and importance. First, user clicks can be considered as implicit feedback on the relevance of documents. Therefore, they can be used as external tags of web pages as well. In addition, the browse logs can be leveraged for calculating the importance of web pages. Intuitively, if a web page is visited by many users, and users stay long on the page, likely the page is important. Section 3 will survey how search and browse log mining can improve web search through query annotation and page importance calculation.

1.3.4 *Enhancing document ranking.* The key functionality of a web search engine is to rank documents according to their relevance to a given query. It has been well studied in traditional information retrieval how to leverage users' explicit feedbacks to improve the quality of ranking. Search logs provide users' implicit feedbacks on ranking. Intuitively, if users often skip a search result ranked at a higher position but click on a result at a lower position, then the lower ranked result is likely to be more relevant than the higher ranked one.

In Section 4, we will review two approaches to mining click-through data. The first approach derives information on the relative relevance of document pairs. The second approach builds a sequential click model to characterize user's click behavior on the list of search result.

1.3.5 *Enhancing user understanding.* Search consists of three major factors: queries, documents, and users. Traditionally, the modeling of users was based on the contents created by the users. The availability of search and browse log data makes it possible to conduct better user modeling and in sequel improve search quality. For example, with better understanding of users, one can conduct disambiguation of queries more effectively, and thus enhance the performance of document ranking.

There are two popular approaches to addressing search using user modeling, namely, *personalization* and *contextualization*. The personalization approach often mines search and/or browse log data and creates a user profile for each individual user to characterize her preference, as will be described in Section 5.1. The contextualization approach creates a context profile from log data to capture users' preference in different contexts, as will be explained in Section 5.2.

1.3.6 *Monitoring and predicting user satisfaction.* Obviously, search and browse log data are useful for evaluation of a search system. Numerous metrics, such as numbers of queries, numbers of users, and average click-through rates can be calculated from log data with respect to different markets and time periods. In the meantime, various models can be applied to predict user satisfaction from search and browse log data. The applications of log data for monitoring and predicting user satisfaction will be discussed in Section 6.

1.3.7 *Enhancing efficiency of web search.* Mining search and browse log data can enhance the efficiency of web search [Silvestri 2010]. For example, search logs can be exploited to build a caching mechanism to reduce the workload of search engine. The data can also be leveraged to make a balanced partition of documents and terms within the distributed system of search engine.

The remaining part of this survey is organized as follows. From Sections 2 to 6, we will elaborate how search and browse log data can enhance each of the five major components of a search system, namely query understanding, document understanding, document ranking, user understanding, and monitoring and feedbacks. Section 7 gives a summary of the survey.

2. QUERY UNDERSTANDING

Query understanding aims at understanding intent of queries, performing better document ranking, and providing better search result presentation. It is also useful in tasks such as query completion (for example, [Shokouhi and Radinsky 2012]) and query suggestion (for example, [Ozertem et al. 2012]). Obtaining statistics about queries often serves as the first step. In addition, most of the existing work addresses query understanding as query classification, query segmentation, query transformation, and named entity recognition in queries. We describe them in detail.

2.1 Query Statistics

Let us start with some basic statistics of queries, as matter of fact. This will be useful for understanding of existing work.

Numerous studies were conducted to analyze queries in search and browse log data from different perspectives, including (1) how users conduct search, such as query length, query frequency, query term frequency, number of viewed search result pages, session length; and (2) what users search for, such as, topic distribution and function. For detailed analysis on the above two aspects, please see [Hölscher and Strube 2000; Jansen et al. 2007; Jansen and Pooch 2001; Jansen and Spink 2006; Spink et al. 2001; Spink et al. 2002; Silverstein et al. 1999; Silvestri 2010; Wolfram et al. 2001]. The major conclusion from the statistics is that web search is very different from traditional information retrieval. For other statistics on queries, see [Beitzel et al. 2004; Beitzel et al. 2007; Backstrom et al. 2008; Mei and Church 2008; Weber and Jaimes 2011; Weerkamp et al. 2011].

The average length of search queries is about 1.66-2.6 words, which is much shorter than in traditional IR (6-9 words) (cf.,[Jansen et al. 2007]). The average length is becoming longer, but in general remains constant in a relatively short time period.

Query frequencies and query term frequencies follow power law distributions. That is, head (high frequency) queries and terms account for the majority of search traffic. On the other hand, tail (low frequency) queries and terms also consist of large percentages of the distributions.

It is also observed that users browse on average less than two web pages in search results and over half of the users do not access results beyond the first pages. This observation implies that the relevance of the top ten results is critically important for a web search engine. Average session length is around two to three queries and

more than half of the sessions consist of only one query.

The major topical categories of queries are "people and place", "commerce", "health", "entertainment", "internet and computer", and "pornography". The relative order among them can be slightly different over time and in various regions.

There are four major types of queries from the viewpoint of linguistic structure [Bendersky and Croft 2009], namely (1) noun phrases, (2) composition of noun phrases, (3) titles (titles of books, products, music, etc), and (4) natural language questions. The majority of queries fall into the first two types.

2.2 Query Classification

Queries can be classified along multiple dimensions including search goals, topics, time-sensitivity, and location-sensitivity. The classes of a query can be used to represent the intent of the query.

All types of search and browse log data can be used for query classification, in addition to web data. In fact, click-through data, session data, and search result data are widely used for the query classification tasks.

Query classification, however, is a challenging task, due to the following reasons: (1) queries are short, (2) queries are usually ambiguous, (3) the meanings of queries may change over time and location, (4) queries are also noisy, for example, about 10% of queries contain typos.

2.2.1 *Search Goals.* Broder [2002] and Rose and Levinson [2004] pointed out that, from the viewpoint of the goals of search in the searchers' minds, search intents can be categorized into navigational intents, informational intents and transactional intents. A navigational search is to reach a particular site, an informational search is to acquire information assumed to be present on one or more web pages, and a transactional search is to perform some web-mediated activity. In a specific search, the intent of the user in terms of search goal should be very clear, either navigational, informational, or transactional.

Queries are often ambiguous. The same query can be used in different searches to represent different search goals. However, it is often the case that for a specific query there is a dominating intent. Then, given a query, is it possible to identify its dominating search goal? This is the problem of query classification by search goal.

Lee et al. [2005] proposed to use distribution of clicks and that of anchor text to identify whether the dominating intent of a query is navigational or informational, assuming that transactional is included in navigational. Their method mainly takes advantage of the following heuristic in click-through data and web data to perform the task. If a query is navigational, then main search intent represented by the query is to find a specific web page. Thus, the majority of clicks with respect to the query should be on a single URL in the click-through data. For example, for the query "microsoft", the majority of clicks are on the URL `microsoft.com`. If a query is navigational and also exists in the anchor texts in a web collection, then the majority of the anchor texts should be linked to the same URL. For example, the majority of the anchor texts "Microsoft" should point to the URL `microsoft.com`. In other words, the click distribution of a navigational query is skewed, and so is the anchor distribution of a navigational query.

They used the above information as features to train a classifier to identify whether a given query is more likely to be navigational or informational. The information is specifically represented as mean, median, skewness, etc. They asked human subjects to manually label queries as navigational or informational and took majority votes on the labeled results. The accuracy for the classification was about 90% on the labeled data. The challenge for the approach is that it is difficult to be applied to tail queries because much less click data and web data are available for tail queries.

Queries can also be classified by search tasks [Ji et al. 2011; Liao et al. 2012].

2.2.2 *Semantic class.* Query classification by their semantic classes or topics is useful in search result presentation. The classified semantic classes of a query can also be included in the query representation.

The classes in query classification can be coarse (for example, 'internet') or fine-grained (for example, 'NBA', 'shoes'). Shen et al. [2006] designed a method for the former case, while Fuxman Fuxman et al. [2008] and Li et al. [2008] tackled the latter case.

One simple approach to query classification is to view a query as a short text and formalize query classification as text classification. Since queries are short and ambiguous, this approach does not work well. To enhance the accuracy of classification, Shen et al. [2006] and Shen et al. [2006] proposed to use the search results of a query at web search engines to enrich the original query and use the enrichment of the query in classification. Specifically, in their method each query is submitted to several web search engines. In the search result of each search engine, the semantic categories of the web pages, as well as the titles and snippets of the web pages are collected. The former is viewed as synonyms and the latter is viewed as pseudo texts of the query. Two classifiers are then applied to the synonyms and pseudo texts respectively for query classification. The classifiers from different search engines are then linearly combined to make the final classification decisions. Shen et al. [2006] used the techniques and won the championship at the query classification task of KDD Cup 2005.

Fuxman et al. [2008] proposed a method for classifying queries using a click-through bipartite graph. Their method views a click-through bi-partite as an undirected graph. In the graph, there is a special node called the null node. Every other node is linked to the null node. A random walk model on the graph is then defined. In the model, the probability of an edge represents the transition probability, which is calculated based on click counts, the probability of a node represents the probability of belonging to a class. In binary classification, some nodes (query nodes, URL nodes, or both) are labeled as positive examples at the beginning. That is, their probabilities of being in the class are all set to one. The learning process of the model then becomes prorogation of the class labels to the rest of nodes on the graph through random walk. Fuxman et al. [2008] adopted an efficient iterative algorithm to conduct the label propagation. The algorithm keeps updating the probabilities on the nodes and is guaranteed to be converged. Finally, the unlabeled nodes are assigned probabilities of belonging to the class. The model is an analogy to an electrical network. The probabilities on the edge correspond to conductance, the probabilities of nodes correspond to voltage and the null node corresponds to the

ground. At the beginning the labeled nodes have a unit of voltage, and after some iterations the voltages of all the nodes become stable.

Li et al. [2008] proposed a method of classifying queries into a class (topic) using click-through bipartite and some labeled queries. The labeled queries include both positive and negative examples. They employed two classifiers for the task. One is based on content, and the other one based on propagation on the click-through bipartite graph. In learning, their method iteratively trains the two classifiers. First, it trains a classifier based on the content of the labeled queries, and then propagates the class labels on the click-through bi-partite. Finally, the content based classifier is used for query classification. In the label propagation classification, their method views the click-through bipartite as an undirected graph and formalizes the learning problem as propagation of the labels from the labeled nodes to the whole graph, assuming that the similarities between nodes are calculated from the click counts. The intuitive explanation is that the class memberships of unlabeled nodes can be inferred from those of the labeled nodes according to the proximities of the nodes in the graph. Their method is equivalent to increasing the amount of training data by semi-supervised learning with click-through bipartite graph. They showed that with their method it is possible to significantly increase the training data size, and thus significantly improve the accuracy of content based query classification.

For other methods of query classification, please see also [Baeza-Yates et al. 2006; Beitzel et al. 2007; Hu et al. 2009; Kang et al. 2011; Hu et al. 2012].

2.2.3 *Location Sensitivity.* Queries may also have location intents, which means that the searchers want to get information close to certain locations, such as the search locations. For example, when a user submits a query "pizza hut", she may want to find the information about the Pizza Hut restaurants nearby. According to Welch and Cho [2008], about 30% of queries are location sensitive, which are called localizable queries. Welch and Cho [2008] also proposed a classifier for identifying whether a query is a location sensitive query. Their method makes use of features derived from search log data and a dictionary of locations. The method takes advantage of the fact that a location sensitive query is likely to co-occur with location names in other queries. For example, the location sensitive query "car rental" may co-occur with many location names, such as "california" and "new york". The features used by the classifier include whether the query frequently co-occurs with location names in the log data and whether the distribution of co-occurrence between the query and the location names is close to uniform. For example, "Car rental" is a location sensitive query, since it occurs with many different location names. In contrast, although "declaration" frequently co-occurs with the location name "Independence", a city in Missouri, in the queries "declaration of independence", it is not a location sensitive query. This is because the distribution of the query with location names is skewed. For query classification based on location, please see also [Yi et al. 2009].

2.2.4 *Time Sensitivity.* Queries may have time intents as well. When submitting a time sensitive query, a searcher often intends to find information in a time period, typically very recent. In the context of recency ranking in web search, Dong et al. [2010] proposed a method for identifying whether a query is time sensitive,

specifically recency sensitive. The key idea is to calculate the difference between the likelihood of a query in the current time slot and that in the past time slots both in query logs and news articles. If there is a significant difference, then it is likely that the query is recency sensitive. More specifically, the method creates language models from query logs and news articles (contents) in the current time slot and the historical time slots of past day, past week, and past month, referred to as $M_{Q,t}$, $M_{C,t}$, $M_{Q,t-r_i}$, and $M_{C,t-r_i}$ respectively, where i denotes the nested past time slots. Given a new query in the current time slot, the method estimates the likelihoods of the query being generated from each of the models. It calculates the buzziness of the query from the query logs as

$$buzz(q,t,Q) = \max_i \{\log P(q|M_{Q,t}) - \log P(q|M_{Q,t-r_i})\}$$

The buzziness of the query from the news articles $buzz(q,t,C)$ is calculated similarly. The final score is obtained by linearly combining the two buzziness scores.

$$buzz(q,t) = \lambda \cdot buzz(q,t,Q) + (1-\lambda) \cdot buzz(q,t,C)$$

For other related work, please see [Vlachos et al. 2004; Chien and Immorlica 2005; Kulkarni et al. 2011].

2.3 Query Transformation

Query transformation changes an original query to a query or some queries similar to it, such as changing "ny times" to "new york times". Query transformation is also referred to as similar query finding, query rewriting, and query alteration [Croft et al. 2010]. A query is considered similar to another query if they share similar search intents. Query transformation can be exploited for enhancing search relevance as means of query expansion and query reduction. It can also be used in query suggestion. Query transformation can be performed by using click-through data and session data. It can also be performed by models trained with click-through data and session data.

2.3.1 *Query Transformation Using Click-through Bipartites.* In a click-through bipartite graph, similar queries may have similar clicked URLs associated. The co-click information can be used in finding similar queries. Likewise, the information can also be used in finding similar URLs. We consider two types of query transformation or similar query finding using click-through bipartite graphs. One is to calculate similarity scores and the other is to conduct query clustering.

Xu and Xu [2011] calculated query similarity from click-through bi-partite using Pearson correlation coefficient. They found that when Pearson coefficient is larger than 0.8, more than 82.1% of query pairs are indeed similar query pairs.

Beeferman and Berger [2000] proposed an agglomerative clustering algorithm for clustering similar queries using a click-through bipartite graph. Their algorithm is completely content-ignorant in the sense that it makes no use of the actual contents of the queries and the URLs, but is based on only how they co-occur within the click-through bipartite. Although the algorithm is simple, it can discover high quality query clusters. For other methods of query clustering using click-through, refer to [Cui et al. 2002; Wen et al. 2001].

A click-through bipartite graph in practice can be extremely large. How to

efficiently perform clustering on click-through bi-partite then becomes a critical issue. Cao et al. [2008] developed an efficient algorithm for the task of context aware query suggestion. To significantly enhance the efficiency of clustering, they leveraged the fact that on average each query only has about three associated URLs and each URL only has about three associated queries. The algorithm needs only one scan of the data and the average case time cost is linear to the number of instances. Specifically, their algorithm adopts an agglomerative approach to clustering. It linearly scans the data and incrementally creates clusters. Given the current instance, the algorithm compares it with the centroids of all the existing clusters. If the instance is close enough to one of the clusters, then it is added into the cluster; otherwise, a new cluster is created for the instance. The algorithm takes advantages of the fact that, when calculating the similarity between two instances (also between an instance and a centroid) in Euclidian distance or dot product, one only needs to make calculation on the non-zero elements shared by the two instances. The algorithm creates and maintains an inverted index about all the non-zero elements of the instances in the existing clusters. Once a new instance comes, it only takes its non-zero elements, looks up the index and makes similarity comparison with the clusters that also have non-zero elements in the same positions. Later, Liao et al. [2011] further improved the method such that post-processing is conducted to re-merge and re-split clusters to reduce the low quality clusters due to the order of input.

Another state-of-the-art method for finding similar queries from click-through bi-partite graph was proposed by Craswell and Szummer [2007]. From click-through bi-partite, one can find not only query-query similarity, but also document-document similarity, and query-document similarity. Craswell and Szummer's method takes the click-through bipartite graph as a directed graph and defines a backward random walk model on the graph. The transition probabilities $P(k|j)$ from node j to node k is calculated by normalizing the click counts out of node j, that is,

$$P(k|j) = \begin{cases} (1-s)\frac{C_{jk}}{\sum_i C_{ji}} & k \neq j \\ s & k = j \end{cases}$$

where C_{jk} denotes the number of clicks on the edge between the nodes j and k, and s is the self-transition probability. The random walk usually takes several steps of walk and stops. It is observed that the role of self-transition probability is very important. Craswell and Szuemmer's model is actually a model of similarity weight propagation on the click-through bipartite graph. For other work on finding similar queries from a click bipartite graph, see [Mei et al. 2008].

2.3.2 *Query Transformation Using session data.* Searchers sometimes issue similar queries in the same search sessions. We can also find pairs of successive queries from search sessions. Jones et al. [2006] developed a method for discovering similar query pairs from session data. Given two queries q_1 and q_2, they conducted likelihood ratio testing to check whether their co-occurrence in search sessions is statistically significant.

$$H_1 : P(q_2|q_1) = p = P(q_2|\neg q_1)$$

$$H_2 : P(q_2|q_1) = p_1 \neq p_2 = P(q_2|\neg q_1)$$

The likelihood ratio is

$$\lambda = -2 \log \frac{L(H_1)}{L(H_2)}$$

The query pairs whose likelihood ratios are above a threshold are viewed as similar queries or substitutable queries. The similar queries discovered by Jones et al.'s method are based on typical substitutions web searchers make. For other methods of finding similar queries from search session data, please see also [Huang et al. 2003; Fonseca et al. 2005; Boldi et al. 2008; Szpektor et al. 2011].

2.3.3 *Model Based Transformation.* The aforementioned methods can automatically mine similar queries from click-through data and session data. These methods usually work very well for head queries, but not for tail queries. We need to consider using the data from head queries to train models applicable to tail queries. An essential idea is that the linguistic knowledge learned from heads can be applied to tails. For example, if we can learn from head queries "sign on hotmail" and "sign up hotmail" that phrases "sign on" and "sign up" are similar, then we can judge that the tail queries "sign on x-forum" and "sign up x-forum" should also be similar.

Guo et al. [2008] proposed a method based on the above rationale. They viewed query transformation as a mapping from the space of original queries X to the space of refined queries Y. Obviously, directly exploiting the model $P(y|x)$ is not practical, because both X and Y are extremely large, where y and x are random variables taking values from Y and X. They proposed to add another random variable o and employ the model $P(y, o|x)$ to solve the problem, where o takes values from a set of operations. An operation can be insertion, deletion, and substitution of letters in a word, splitting of a word into multiple words, merging of multiple words into a single word, word stemming, or some others. To be specific, the number of mappings from any x in X to any y in Y can be very large. However, the number of mappings from x to y under operation o will be drastically reduced. They defined $P(y, o|x)$ as a conditional random field (CRF) model on query word sequences. They developed methods for learning the model and making prediction using dynamic programming. Given a sequence of query words, the CFR model predicts a sequence of refined query words as well as corresponding refinement operations. One merit of this approach is that different types of transformations, such as spelling error correction, merging, splitting, and stemming, can be performed simultaneously and thus the accuracy of transformation can be enhanced, because sometimes the transformations are interdependent. The data for training the CFR model can be mined from session data using a method developed by Jones et al. [2006].

Spelling error correction in query can be viewed as a specific task of query transformation. Normally, about 10% of queries contain spelling errors and thus spelling error correction is a very important component for web search [Guo et al. 2008]. Guo et al. [2008] developed a discriminative approach. Duan and Hsu [2011] proposed generative approaches to spelling error correction. See also [Li et al. 2012].

2.4 Query Segmentation

A query q can be viewed as a sequence of words $(w_1, w_2, \cdots, w_k)$. A segmentation of query q is a sequence of phrases that can compose the query. For a query of k words, there are 2^{k-1} possible segmentations. Query segmentation is a difficult task, because queries are short and ambiguous. For example, the query "new york times square" may have different segments, "(new york) (times square)" and "(new york times) (square)". Both supervised and unsupervised approaches are proposed for query segmentation.

Bergsma and Wang [2007] proposed to view the query segmentation task as a problem of making a segmentation decision at each adjacent word pair. In the classification framework, the input is a query and a position in the query, and the output is a segmentation decision at the position (yes/no). In segmentation, $k-1$ decisions are made for a k word query. A binary classifier can be trained for the problem. Features like whether the left word is "the" and part of speech of the left word are used. Bergsma and Wang [2007] verified that a local classification approach works better than a global tagging approach such as one using hidden Markov model. The reason is that query segmentation is not a sequential modeling problem.

Hagen et al. [2011] proposed an unsupervised approach to query segmentation. The advantage of unsupervised approach is that no training is needed. Although the method proposed by them is very simple, it works very well in experimentations. The method, called naïve normalization, calculates a score for each segmentation of a query, ranks the segmentations based on their scores, and takes the segmentation with the highest score as output. That is,

$$score(S) = \begin{cases} \sum_{s \in S} |s|^{|s|} freq(s) & \forall s, freq(s) > 0, |s| \geq 2 \\ -1 & \text{otherwise} \end{cases}$$

where S denotes one segmentation and s is a segment (an n-gram) within S, $freq(s)$ is the frequency of s calculated from a large web corpus. The summation is only taken from the segments of more than one word. Furthermore, $|s|^{|s|}$ is a weight favoring long segments (n-grams), because longer segments are more preferable. For example, "toronto blue jays" should not be further segmented to "blue jays", though the latter has a larger frequency than the former.

For query segmentation, see also [Li et al. 2011; Hagen et al. 2012].

2.5 Named Entity Recognition in Queries

Many queries contain named entities in different types, such as personal names, locations, organizations, and product names. Named entity recognition in queries is helpful for search result presentation.

Paşca [2007] and Paşca and Alfonseca [2009] conducted a series of research on the problem and proposed several methods for the task. Their basic idea is to use the patterns of attributes of entities in a class to identify new entities pertaining to the class. Their approach employs weakly supervised learning by assuming that there are a small number labeled instances available, that is, seed entities belonging to a class. It starts the mining process with the seed entities. It matches the seed entities to the query log, discovers context patterns of the entities, and mines new

entities with the patterns. Finally, it calculates the context similarities between the mined entities and the seed entities, and ranks the new entities based on their context similarity scores. For example, "vioxx" is a seed entity of the class Drug. From the query log, its context patterns can be found, such as, "long term * use", "side effect of *", where * is a wildcard. With the patterns, new entities of the class, such as "viagra" and "phentermine", can be mined. Their methods all assume that if A is a prominent attribute of class C and I is an instance of class C, then a fraction of queries about I should be about both I and A.

One challenge for the above deterministic approaches is that it is hard to deal with ambiguities in named entities. For example, "harry potter" can belong to multiple classes including Book, Game, and Movie. The mined attributes can be those for different classes, and thus it is easy to include noises in the mining process. Guo et al. [2009] proposed a probabilistic approach to tackle the disambiguation problem. The approach is based on Weakly Supervised Latent Dirichlet Allocation (WS-LDA), an extension of conventional LDA. Their method creates a pseudo document for each labeled named entity. It views the contexts of a named entity in search log as words of the document with regard to the entity, and the classes of the named entity are regarded as possible topics of the document. Thus, the possible topics for each document are reduced to a small number, though they are still ambiguous. This is a different setting from the conventional LDA. Their method learns the topic model given the labeled data using variational EM algorithm. In learning, the supervision information is incorporated into the objective function as constraints. As a result, the probabilities of words given topics, that is, probabilities of contexts given classes are learned from the documents, and they can be utilized as patterns for entity mining. Since the framework for mining is probabilistic instead of deterministic, more accurate context patterns can be learned by the approach.

3. DOCUMENT UNDERSTANDING

Document understanding is to represent and measure documents (web pages) in an effective way so that documents that are relevant to a query can be retrieved and ranked high in search. Web search takes into account two important aspects of webpages: representation and importance. In this section, we explain how log data may help to improve document understanding in creation of webpage representation as well as calculation of webpage importance.

3.1 Document Representation

A webpage is actually a text document, and can be represented as a vector of TF-IDF scores of the words in it. The vector can then be indexed in a search system and used in the vector space model, the BM25 model, or language model for IR for relevance ranking. This is a conventional way and has been well explored in traditional information retrieval [Salton et al. 1975; Baeza-Yates and Ribeiro-Neto 1999].

Webpages contain hypertexts and thus are more than just words. There is rich information on the web, which can help to enhance the representations of webpages. For example, the *anchor text* of a webpage, pointed from another webpage, often gives a compact and precise description of the page. The anchor texts of a webpage actually reflect how the authors of the other pages on the web think about the

webpage. It has been widely verified that anchor texts are useful information for representing webpages in order to make them better searched.

While anchor texts may help represent webpages, they only represent the views from web content creators, not necessarily web users. Furthermore, the distribution of anchor texts also follows the zipf's distribution, and the tail webpages (unpopular) webpages usually do not have enough anchor texts. Thus, queries as annotations are explored.

3.1.1 *Queries as Annotations.* Since search log data, particularly, click-through data, record user queries and the corresponding clicked webpages, a natural idea to use log data to enhance webpage representations is to use queries as annotations of the webpages. If a user asks a query q and clicks a webpage p in the result list, then it is likely that p is relevant to q in one way or another. Consequently, q can be used as an annotation of page p. In other words, the log data can be regarded as users' annotations of webpages. Although click-through data contain noise, it has been verified that the use of click-through data as annotations of webpages can help to improve relevance significantly [Agichtein et al. 2006; Agichtein et al. 2006]. Obviously, data cleaning, for example, using frequency cut-off, is necessary.

One advantage of using log data to represent webpages is that such data reflect the views on web pages aggregated from many users, which might be more useful for search. Moreover, as log data are accumulated, the annotations from the users will be dynamically and continuously updated.

Poblete and Baeza-Yates [2008] developed two query-based webpage representation models using click-through data. The first one is called query-document model. The major idea is to use the terms in queries associated with a web page as annotations of the page. The query-document model uses a vector of query terms weighted by TF-IDF. The frequency of a term is calculated as the frequency of the queries containing the term associated with the page.

A drawback of the query-document model is that it assumes terms are independent from each other. In practice, some query terms frequently occur together in the same query, expressing a concept. For example, queries "*apple computer*" and "*apple juice*" carry completely different meanings for the word "*apple*".

To deal with the problem, the authors proposed the second model, the query-set model. The model uses frequent query term sets as annotations. Given a set of queries associated with a web page, where a query may contain multiple terms, we can find frequent 1-, 2-, and 3-term combinations in the set of queries based on certain supports (frequency thresholds). Each frequent term combination is called a relevant set. Then, the webpage is represented by a vector of relevant sets weighted by TF-IDF. The term frequency of a relevant set is the number of times that the relevant set appears in the queries associated with the webpage.

3.1.2 *Coping with Data Sparseness.* While using queries to annotate webpages is an intuitive idea, there are several technical challenges. Click-through data are sparse. Many webpages may have very few or even no clicks. For example, Gao et al. [2009] reported that, in a real data set of 2.62 million query-document pairs, 75% of them do not have any click. How can we effectively annotate those webpages? Several studies addressed the challenge using various techniques.

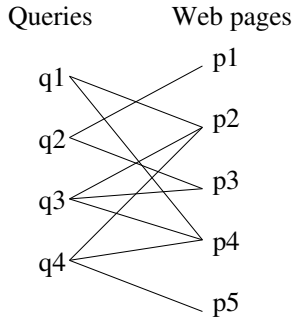

Fig. 7. An example of click through bipartite graph where nodes stand for queries and web pages.

Xue et al. [2004] proposed two methods to deal with the sparseness challenge. The first method makes use of not only the queries associated with a webpage as the representations of the web page, but also the queries associated with its similar webpages. Here, two webpages are considered similar if they are clicked in the searches of the same set of queries. For example, consider the queries and the webpages in Figure 7. A query and a webpage is linked by an edge if the webpage is clicked when the query is searched. Pages p_2 and p_4 are clicked by the same set of queries, namely $\{q_1, q_3, q_4\}$. Thus, p_2 and p_4 are considered similar because they may satisfy the similar information needs. The term frequency of query q_j with respect to webpage d_i is calculated by $W(d_i, q_j) = \sum_{d_k \in Sim(d_i)} S(d_i, d_k) W(d_k, q_j)$, where $Sim(d_i)$ is the set of webpages similar to webpage d_i, $S(d_i, d_k)$ is the similarity between webpages d_i and d_k calculated based on co-click relations, and $W(d_k, q_j)$ is the frequency of query q_j with respect to webpage d_k.

The second method is an iterative approach that mutually reinforces query and document similarities on the click-through bi-partite graph. The method performs random walk on the click-through graph. It is assumed that two webpages are similar if they can be searched by similar queries, and two queries are similar if they can search similar webpages.

Technically, let $S_Q[q_1, q_2] \in [0, 1]$ be the similarity between two queries q_1, q_2 in question, and $S_P[p_1, p_2] \in [0, 1]$ be the similarity between two webpages p_1, p_2 in question. The following equations are used to implement the above mutual reinforcing rules.

$$S_Q[q_1, q_2] = \begin{cases} 1 & \text{if } q_1 = q_2 \\ \frac{C}{|O(q_1)||O(q_2)|} \sum_{i=1}^{|O(q_1)|} \sum_{j=1}^{|O(q_2)|} S_P[O^i(q_1), O^j(q_2)] & \text{otherwise} \end{cases} \quad (1)$$

$$S_P[p_1, p_2] = \begin{cases} 1 & \text{if } p_1 = p_2 \\ \frac{C}{|I(p_1)||I(p_2)|} \sum_{i=1}^{|I(p_1)|} \sum_{j=1}^{|I(p_2)|} S_Q[I^i(p_1), I^j(p_2)] & \text{otherwise} \end{cases} \quad (2)$$

where C is a decaying factor that is set to 0.7 empirically, $O(q)$ is the set of webpages associated with query q, $O^i(q)$ is the i-th webpage in the set $O(q)$, $I(p)$ is the set of queries associated with webpage p, and $I^i(p)$ is the i-th query in the set $I(p)$.

Since Equations (1) and (2) are recursive, we can propagate the similarities

through an iterative process. We start with

$$S^0(p_1, p_2) = \begin{cases} 1 & (p_1 = p_2) \\ 0 & \text{otherwise} \end{cases}$$

and compute S^{i+1} from S^i. It is easy to see that the values S^k are non-decreasing and have an upper bound of 1. Thus, the iterative process converges.

The similarity propagation can overcome the sparsity in log data. Even two webpages p_1 and p_2 do not share any queries, that is, $I(p_1) \cap I(p_2) = \emptyset$, they may still be considered similar if many pages associated with the queries in $I(p_1)$ and $I(p_2)$ are similar.

To overcome the sparseness problem, Gao et al. [2009] proposed a random walk method to smooth the frequencies (weights) of queries. Technically, we can construct the matrices $A_{ij} = P(d_i|q_j)$ and $B_{ji} = P(q_j|d_i)$, where d_i is a web page, q_j is a query, and $P(d_i|q_j)$ and $P(q_j|d_i)$ are the conditional probabilities estimated from the click data. To make each query or document similar to itself, a self-transition is added to each query or document. Then, we can conduct random walk on the click-through graph by multiplying the matrices. Take the click-through bipartite graph in Figure 7 as an example. Without the expansion, webpage p_1 is only associated with query q_2. After a random walk step, the association between webpage p_1 and query q_3 is augmented, because query q_3 has a similar click pattern as query q_2.

3.2 Document Importance

Calculating the importance of web pages is an essential task in web search. Important web pages should be ranked high in the answer list of web pages to a query. Traditionally, the importance of web pages is calculated by link analysis. A link from one page p_1 to another page p_2 is regarded as an endorsement of p_2 from p_1. The more links pointed to a page, the more likely the page is important. The importance of pages can be propagated in the web graph where vertices are pages and edges are hyperlinks. For example, in PageRank [Page et al. 1999], a famous link analysis method, page importance is modeled as a stationary distribution of a Markov chain (random walk) model defined on the web graph. The PageRank scores of web pages can be calculated iteratively. In other words, PageRank is a discrete-time Markov process on the web graph.

HITS [Kleinberg 1999] is another link analysis method to model importance of web pages. Two scores are calculated for each page: a hub score and an authority score. A page has a high hub score if it links to many pages. A page has a high authority score if it is pointed by many pages. Using the heuristic that good hubs tend to link to good authorities, and vice versa, the hub scores and the authority scores can be updated in an iterative way, until they are stabilized for all pages.

While link analysis, such as PageRank and HITS, has been shown effective, one drawback is that those methods only model the importance of web pages from the web page authors' point of view, instead of from the web page users' point of view. Mining user feedback from log data can help to improve the modeling of web page importance.

Liu et al. [2008] proposed an algorithm called BrowseRank, which computes page importance from a user browsing graph built from users' browsing history. In the

user browsing graph, web pages are represented as vertices and the transitions between pages are represented as edges. The staying time on web pages is also considered. A continuous-time Markov process is defined on the user browsing graph and its stationary probability distribution is computed as the page importance. Liu et al. [2008] reported through their empirical study that BrowseRank achieves good performance in several tasks. For example, in web spam fighting, BrowseRank can push many spam websites to the tail buckets. This is because users more frequently browse and spend more time on high quality pages than spam pages. This user behavior is effectively utilized by the algorithm.

Session data representing user browsing trajectories can also help the calculation of page importance. Zhu and Mishne [2009] viewed a session as a sequence of hops through the web graph by a user, and computed ClickRank as the importance of each web page in the session. The ClickRank of a page in a session is defined as a function of the order of the page in the session and the dwell time on the page by the user. Intuitively, the lower the rank order and the longer the dwell time, the more important the page is in the session. The importance score of a page is then calculated as the sum of its scores over all the sessions containing the page.

4. DOCUMENT RANKING

In web search, given a query, a ranking list of webpages is returned, where the webpages are ranked in the descending order of the degrees of relevance to the query (that is, degrees of matching to the query). A critical issue here is to properly rank the webpages with respect to the query on the basis of relevance [Salton et al. 1975; Baeza-Yates and Ribeiro-Neto 1999; Croft et al. 2009]. Log data mining can help to substantially improve this document ranking process. First, we can derive preference on webpages with respect to queries from click-through data. Second, we can use the preference information to improve search relevance, for example, using it as training data or feature in learning to rank [Li 2011]. In this section, we review representative methods for mining preference information from click-through data, in particular, preference pairs and click models.

4.1 Click-through Data as Implicit Feedback

In web search, when a ranking list of webpages is presented to a user, some additional information about the webpages is also provided, such as the titles and snippets of webpages. Then, the user clicks on a webpage in the result list, if she thinks the page looks interesting and relevant to the query, possibly hinted by the snippet. After the user clicks on a webpage in the result list, she may click on another webpage. What may such a click tell us? Possibly, the webpage clicked previously does not completely satisfy the user's information need. Therefore, the user is looking for more information. The webpage she clicks later may be more relevant.

As we can see, users' click-through activities can be regarded as users' implicit feedback about the search results. That is, when a user clicks a webpage, the user does not explicitly tell whether the webpage satisfies her information need, or to what extent the page satisfies. The click-through activities, however, provide hints about the users' preference on the webpages with respect to queries.

Now, the question is how we can use the users' click-through data to derive preference information between queries and documents and how to use the preference information to improve search relevance. Under the assumption that a click of a webpage implicitly suggests that the webpage is relevant, a naïve method is to promote those webpages clicked in the searches of a query and demote those webpages un-clicked. However, such a naïve method has some fatal drawbacks. First, there exists position bias from the ranking list of webpages. Users usually scan the webpages from the top. Those webpages ranked at higher positions may have a better chance to be clicked. A webpage hidden at a late position, say the 1000-th position in the ranked list, would unlikely be viewed by the users, even if it is perfectly relevant to the query. Second, due to many reasons, relevant documents may not be included in the ranking list to the query. The naïve method does not work in such a situation.

We need to employ more appropriate methods to learn users' preference on webpages from click-through data. There are two ways to model user preferences. We can capture the pair-wise preferences. That is, given webpages a and b, we try to learn from the click-through data which one is more preferable. Alternatively, we can learn the user preference order on a set of webpages. Correspondingly, the methods of using log data to improve ranking relevance can be divided into two groups, namely the preference pair methods and the click models.

4.2 Preference Pairs

For a query q, suppose a search engine returns a ranked list of webpages $\langle d_1, \ldots, d_n \rangle$, and a user clicks some webpages in the list. A brute-force method to learn preferences is to assume that the clicked webpages are more preferable to those not clicked. That is, we can extract a preference relation $d_i \prec d_j$ for $1 \leq j < i$, when d_i is clicked, and d_j is not clicked, meaning d_i is more preferable to d_j. Such a brute-force method, however, leaves much information unused. Importantly, no preference is derived between any two clicked webpages. Similarly, no preference is derived between any two non-clicked webpages, either. We need a systematic way to model preference pairs.

Joachims [2002] and Joachims et al. [2005] examined individual users' implicit feedback in some click-through data. In a ranked list of webpages $\langle d_1, \ldots, d_n \rangle$ with respect to a query q, let C be the set of clicked webpages. They suggested and verified using real data that the following types of preferences can be extracted.

—A clicked page is more preferable to the pages skipped above, that is, $d_i \prec d_j$ for all pairs $1 \leq j < i$ with $d_i \in C$ and $d_j \notin C$. As a variant, the last clicked page is more preferable than all the pages skipped above. Their experimental result shows that the variant is slightly more accurate than its general form.

—A clicked page is more preferable to the pages clicked earlier, that is, $d_i \prec d_j$ for all pairs with $d_i, d_j \in C$ and $t(d_i) > t(d_j)$, where $t(d_i)$ is the time when d_i is clicked. In this way, we can derive preferences among clicked webpages.

—A clicked page is more preferable to the next page in the list if it is not clicked, that is, $d_i \prec d_{i+1}$ for all $d_i \in C$ and $d_{i+1} \notin C$.

Joachims [2002] proposed a method for enhancing relevance ranking using the

preferences learned from click-through data. More specifically, he trained a Ranking-SVM model using the preference pairs as training data.

Dou et al. [2008] compared the effectiveness of using preferences derived from click-through data and using human labeled data to train a learning-to-rank model. Let $click(q, d)$ be the aggregated click frequency of webpage d with respect to query q. For a query q, to derive the preference between a pair of webpages d_1 and d_2, one can calculate $cdif(q, d_1, d_2) = click(q, d_1) - click(q, d_2)$. If $cdif(q, d_1, d_2) > 0$ (or a threshold), then d_1 is regarded more preferable to d_2 for query q. They found that the preferences derived from click-through data can be used as training data for learning-to-rank with the advantage of low cost.

In practice, a user may not represent her information need perfectly at the first place, and she may reformulate her query and conduct search again with the new query. Therefore, we can use a sequence of queries, called a query chain, and the corresponding clicks in a search session by a user as an instance in learning of preferences. Radlinski and Joachims [2005] proposed several rules to extract user preferences from query chains and the corresponding click-through data. Those rules are extensions of the methods by Joachims et al. [2005]. The essential idea is that a user may likely look for the same information using two queries in the same query chain.

Joachims et al. [2005] reported that the probability of a webpage in a ranked list being clicked is heavily biased toward higher positions in the ranked list, known as position bias. Position bias may strongly affect the effectiveness of pairwise preference learning. Thus, it is important to develop position bias free methods for the learning task. Radlinski and Joachims [2006] gave a simple FairPairs algorithm as follows.

Let $R = \langle d_1, \ldots, d_n \rangle$ be the ranked list of webpages for a query. The FairPairs algorithm randomly draws a value $k \in \{0, 1\}$ with equal probability. If $k = 0$, then, for all odd numbers i ($1 \leq i \leq n$), swap d_i and d_{i+1} in R with probability 0.5. Similarly, if $k = 1$, then, for all even numbers i ($1 \leq i \leq n$), swap d_i and d_{i+1} in R with probability 0.5. Then, the list R with the above changes is presented to the users and the user clicks are recorded. When the pairwise preferences are extracted, for a swapped pair d_i and d_{i+1} in which d_i is clicked, d_i is regarded more preferable to d_{i+1}. The authors theoretically proved that the preferences extracted in this way are unbiased toward higher positions.

In addition to position bias, one issue in learning preferences is that very often users only consider the top ranked webpages and seldom evaluate the webpages at low positions, such as those outside the first page. Consequently, the click-through data recorded by a search engine in a passive way are strongly biased toward webpages that are already ranked high. Those webpages highly relevant to a query but initially ranked low may never be viewed or evaluated by any users. To overcome this problem, Radlinski and Joachims [2007] proposed an active exploration approach. A naïve method may intentionally put unevaluated webpages in the top positions. Those unevaluated webpages, however, may be irrelevant to the query, and thus may seriously hurt user satisfaction. The authors developed a principled approach to overcome the problem using a Bayesian method. The central idea is to present to users a ranked list of webpages that is optimized to obtain user feedback.

4.3 Click Models

Pairwise preferences are relatively easy to learn. Such preferences, however, may not generate a ranked list in general. For example, suppose that for web pages a, b, and c, pairwise preferences $a \prec b$, $b \prec c$, and $c \prec a$ are learned from click-through data. The preferences cannot lead to a ranked list among the three web pages.

To overcome the problem, click models are learned and exploited, which can produce a ranking of web pages for a given query on the basis of the click-through data of the query. In other words, using a click model one can predict the preference of webpages with respect to the query. We review several click models called sequential click models here.

To learn sequential click models, one critical problem is to cope with position bias. As we discussed, the probabilities of clicks are affected by the positions of web pages in the ranking list. Thus, the probability of a webpage being clicked in a sequential click model also depends on the position. Formally, this probability $P(c|r, u, q)$ is the probability that webpage u presented at position r is clicked by a user who issues query q.

Craswell et al. [2008] examined several sequential click models. They used as the baseline the hypothesis that there is no position bias. That is, $P(c|r, u, q) = P(a|u, q)$, where $P(a|u, q)$ is the attractiveness of webpage u given query q.

They considered the examination hypothesis: users examine the webpages before they click and the examinations only depend on the positions of webpages. In this hypothesis, every position is associated with a probability $P(e|r)$ of being examined. Therefore, $P(c|r, u, q) = P(e|r)P(a|u, q)$. Please note that the examination hypothesis is a generalization of the baseline, since the latter can be obtained from the former by setting $P(e|r) = 1$.

Another hypothesis they considered is the cascade model, which assumes that users view search results in a top-down manner. Users make a decision on whether they click a webpage before they move to the next webpage. Under such an assumption, each web page is either clicked with a probability $P(a|u, q)$ or skipped with a probability $1 - P(a|u, q)$. Moreover, a user who clicks never comes back, and a user who skips always continues until she clicks. Thus, we have

$$P(c|r, u, q) = P(a|u, q) \prod_{i=1}^{r-1} (1 - P(a|u_i, q)).$$

Essentially, the cascade model captures the user behavior that sequentially examines all web pages in the result until a relevant web page is found. After the first click, the search is regarded done and the rest of the search result is abandoned.

Craswell et al. [2008] tested those hypotheses on real data, and reported that the cascade model performs significantly better than the other models for prediction on clicks at higher positions, but slightly worse for prediction on clicks at lower positions.

There are several factors that the above models do not consider. First, the relation between a webpage and the webpages clicked so far is an indicator on whether the user is likely to click the webpage. Second, there is no distinction between navigational queries and informational queries. For navigational queries users tend to stop at the most relevant webpages, while for the informational queries

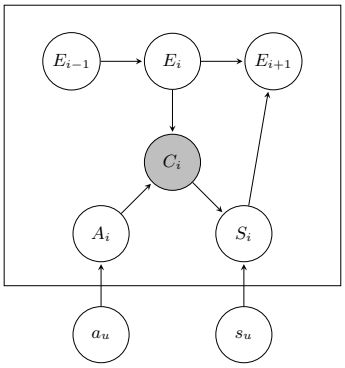

Fig. 8. A user's search behavior can be modeled by the Dynamic Bayesian network model (extracted from [Chapelle and Zhang 2009]).

users tend to click multiple webpages.

To address those issues, Dupret and Piwowarski [2008] examined several user browsing models. In the single browsing model, the probability that a user examines a webpage depends on the distance from that page to the position of the last clicked webpage. This is based on the heuristic that a user tends to abandon a search after she sees less attractive snippets. Mathematically, they modeled both the attractiveness and examination as Bernoulli variables.

Navigational queries and informational queries can be regarded as two extremes in a wide spectrum of queries. Queries may stay at different positions between the two extremes. To address the general varieties of queries, in the multiple browsing model, a mixture of single browsing models is built, and a latent variable is used to indicate which specific single browsing model is used for a particular query. Their experimental results on real data show that the single browsing model has clearly better performance.

Chapelle and Zhang [2009] proposed a dynamic Bayesian network to learn the preferences of webpages with respect to queries from click-through data. Figure 8 gives an illustration of the model. C_i denotes whether a user clicked on a webpage at position i. Variables E_{i-1}, E_i, E_{i+1} denote whether the user examined the webpages at the three positions $i-1, i, i+1$, respectively. A_i denotes whether the user feels that the snippet is attractive and S_i denotes whether the user is satisfied by the webpage. Variables a_u and s_u represent attractiveness and relevance, respectively. C_i is the only observed variable. The variables in the box are repeated from positions 1 to n. The following equations hold in the model.

$$
\begin{array}{ll}
(a) & A_i = 1, E_i = 1 \Leftrightarrow C_i = 1 \\
(b) & P(A_i = 1) = a_u \\
(c) & P(S_i | C_i = 1) = s_u \\
(d) & C_i = 0 \Rightarrow S_i = 0 \\
(e) & S_i = 1 \Rightarrow E_{i+1} = 0 \\
(f) & P(E_{i+1} = 1 | E_i = 1, S_i = 0) = \gamma \\
(g) & E_i = 0 \Rightarrow E_{i+1} = 0
\end{array}
\tag{3}
$$

The dynamic Bayesian network model can closely mimic the user's search behavior. It is assumed that a user clicks a webpage if and only if she looks at the snippet and thinks that it is attractive (Equation 3(a)). The probability of being attracted depends only on the snippet of web page (Equation 3(b)). The user is assumed to sequentially scan the list of webpages from the top to the bottom until she decides to stop. If the user clicks and visits the webpage, then there is a probability that she will be satisfied with the webpage (Equation 3(c)). On the other hand, if she does not click, then she will not be satisfied (Equation 3(d)). Once the user is satisfied, she will stop the search (Equation 3(e)). If the user is not satisfied with the webpage, then there is a probability that the user will abandon the search (Equation 3(f)) and there is another probability that the use will examine the next snippet. If the user does not examine the snippet at position i, then she will not examine the subsequent positions (Equation 3(g)).

Some recent studies further improve the click models, such as [Liu et al. 2009; Guo et al. 2009; Guo et al. 2009; Wang et al. 2010; Hu et al. 2011].

5. USER UNDERSTANDING

We describe two tasks in user understanding, namely, *personalization* and *contextualization*. Personalization typically builds a user profile to describe each individual user's preference, while contextualization creates a context profile to capture the environment of a search activity. In both cases, log mining plays an important role.

Some researchers consider contextualization as a special case of personalization (for example, [Dou et al. 2007]), because contextualization only takes into account users' short search history, while others consider personalization as a special case of contextualization (for example, [Pitkow et al. 2002; Wedig and Madani 2006]), because a user can be viewed as a part of the context in contextualization.

5.1 Personalization

A personalization method usually consists of two steps. First, it constructs a profile for each user using the content data as well as log data. Next, when a user u issues a query, it extracts the profile of the user and applies it into the ranking function. Various methods proposed for creating user profiles can be divided into three major categories, namely, *click-based profiles*, *term-based profiles*, and *topic-based profiles*.

5.1.1 *Click-based Methods.* Teevan et al. [2007] showed that users often repeatedly visit the same web pages by conducting searches at search engines. In other words, users submit the same queries and click on the same search results. Click-based personalization methods take into account the fact, and promote the rank of a page p with respect to query q for user u if evidence from search log data shows that page p is often searched by user u with query q. See also [Bennett et al. 2012].

For example, Dou et al. [2007] defined the following click-based ranking score,

$$S(q,p,u) = \frac{click(q,p,u)}{click(q,\cdot,u)+\beta}, \tag{4}$$

where $S(q,p,u)$ is the personalized relevance score of document p with respect to query q and user u, $click(q,p,u)$ is the number of times user u clicks on document p with respect to query q in the log data, $click(q,\cdot,u)$ is the total number of times

user u clicks on documents with respect to query q, and β is a smoothing factor. The more times document p is clicked by user u with respect to query q, the higher personalized relevance score p receives. In practice, the above ranking model does not work for new queries, and suffers from data sparsity.

To address the problem of data sparsity, Dou et al. [2007] proposed to borrow the idea of collaborative filtering and use the information from other users to conduct smoothing on the relevance scores. If user u' who is similar to user u searches with query q before, then the click information from user u' can be leveraged to estimate $S(q, p, u)$. Let function $sim(u_s, u)$ represent the similarity between two users u_s and u, the personalized relevance score in Equation (4) can be redefined as

$$S(q, p, u) = \frac{\sum_{u_s} sim(u_s, u) click(q, p, u_s)}{\beta + \sum_{u_s} click(q, \cdot, u_s)}. \quad (5)$$

It means that the more similar user u_s is to user u, the more likely the clicked pages by u_s are also clicked by u, and thus the higher relevance score document p has. The similarity function $sim(u_s, u)$ in Equation (5) can be defined in different ways. Dou et al. [2007] classified web pages into pre-defined topics and learned each user's preference on the topics using the pages visited by the user. The similarity between users is then determined by the similarity between their topic preferences.

Sun et al. [2005] proposed the CubeSVD algorithm, which conducts three-order singular value decomposition on the query-document-user cube. We can also use the algorithm to calculate the relevance score of a document with respect to a query and a user, which turns out to be another click-based method. CubeSVD employs the more general higher-order singular value decomposition (HOSVD) [Lathauwer et al. 2000] and is an extension of Latent Semantic Analysis [Deerwester et al. 1990]. Specifically, the CubeSVD method builds a three-mode tensor $C \in R^{l \times m \times n}$ from the log data, where l, m, and n are the numbers of users, queries, and documents, respectively, and each element C_{uqp} ($1 \leq u \leq l$, $1 \leq q \leq m$, and $1 \leq p \leq n$) denotes the number of times document p is clicked by user u with respect to query q. The method then calculates the core tensor S from C using HOSVD. The core tensor S can capture the major latent factors among the users, queries, and documents in C. Finally, the CubeSVD method derives a new tensor $\hat{C}$ from the core tensor S. An element $\hat{C}_{uqp}$ in the new tensor $\hat{A}$ represents the personalized relevance score of document p with respect to user u and query q. Since the correlation among users is encoded in the core tensor S, even if user u never raises query q, her preference on page p with respect to query q can still be estimated. For other related work see also [Jin et al. 2004].

5.1.2 *Term-based Methods.* Compared with click-based methods, term-based personalization methods are more robust to sparse data. They typically create a profile for each user through documents visited or queries issued by the user, and integrate it into the ranking model BM25 [Jones et al. 1998] or language model [Lafferty and Zhai 2001].

Teevan et al. [2005] created a profile for each user u, consisting of tuples (t_i, w_i^u), where t_i is a term and w_i^u is the weight of term t_i with respect to user u. This profile is then applied into the BM25 model to re-rank search result. The BM25

score of document d with respect to query q and user u is defined as

$$S^u(q,d) = \sum_{t_i \in q} \frac{tf_i(k_1+1)}{k_1 + tf_i} w_i^u,$$

where t_i is a term in query q, tf_i is the term frequency of t_i in document d, k_1 is a constant, and w_i^u is the term weight with respect to user u, calculated in the same way as in relevance feedback

$$w_i^u = \log \frac{(|D_i^u| + 0.5)(N - n_i + 0.5)}{(n_i + 0.5)(|D^u| - |D_i^u| + 0.5)},$$

where N is the total number of documents in the corpus, n_i is the number of documents containing w_i, D^u is the set of documents browsed by user u and D_i^u is the subset of documents in D^u that contain term w_i. Their method assumes that the pages browsed by user u are relevant to u, either explicitly or implicitly judged by the user.

Tan et al. [2006] built personalized language models. Suppose that query q_i is submitted by user u. The method finds in the user's search history H_u all queries q_j that user u asked before. For each query q_j, the method constructs a language model θ_j from both the clicked and unclicked search results of q_j. It then uses the personalized language models in search.

The retrieval framework based on language models is formally defined as

$$D(\theta_i||\theta_d) = \sum_{t \in V} p(t|\theta_i) \log \frac{p(t|\theta_i)}{p(t|\theta_d)}, \tag{6}$$

where θ_i and θ_d, respectively, are the language models for query q_i and document d, $p(t|\theta_i)$ and $p(t|\theta_d)$ are the probabilities of term t based on models θ_i and θ_d, respectively, and $D(\theta_i||\theta_d)$ is the Kullback-Leibler divergence between θ_i and θ_d. The major idea of their method is to replace the query language model θ_i in Equation (6) by the personalized language model θ_i^u, which includes user's history information H_u. The probability distribution of θ_i^u is specifically defined as

$$p(t|\theta_i^u) = \lambda_i p(t|\theta_i) + (1 - \lambda_i) p(t|\theta_i^h),$$

where λ_i is a parameter between 0 and 1, and θ_i^h is the language model constructed from user u's search history H^u. Let $H^u = \{q_1, \ldots, q_k\}$, θ_i^h is defined as the weighted sum of the language models of the queries in H_u, normalized by the sum of the weights. Then,

$$p(t|\theta_i^h) = \frac{\sum_{q_j \in H_u} \sigma_j p(t|\theta_j)}{\sum_{q_j \in H^u} \lambda_j},$$

where the language model θ_j for each query q_j can be estimated from both the clicked and unclicked documents of q_j, and the weight σ_j for model θ_j depends on the similarity between q_i and q_j. The more similar q_j is to q_i, the more influence θ_j has on the personalized model θ_i^u.

5.1.3 *Topic-based Methods.* The term-based methods may not be applicable to a query and a user, if none of the terms in the query occurs in the user's search history. In such cases, we may consider employing topic-based methods.

A topic-based personalization method creates a topic profile for each user. In general, a topic profile π_u for user u is represented by a vector, where each element $\pi_u[c_i]$ indicates the probability that the user is interested in a particular topic c_i. The probabilities are estimated from the user's search and/or browse history. For example, we may collect the terms in the user's queries, clicked documents, and browsed documents, and then apply conventional text classification techniques (for example, [Berry 2003]) to infer the user interests on a set of topics. Once the user profiles are created, different topic-based methods mainly differ in how they integrate the topic profiles into the ranking function.

Pretschner and Gauch [1999] adopted the Magellan hierarchy as the set of topics. The hierarchy consists of 4,400 nodes (that is, topics) and each node is associated with a set of documents. Their method builds a TF-IDF vector $V(c_k)$ for each node c_k using its associated documents. To construct the topic profile π_u for user u, the method first collects all the documents d visited by the user and then estimates the probability $P_d[c_k]$ of each document d belonging to each topic c_k. Finally, the user's preference on topic c_k is derived by aggregating the probabilities $P_d(c)$ over all the visited documents. Given query q, the personalized ranking score for document d with respect to user u is defined as

$$S(q, d, u) = S(q, d) \cdot (0.5 + \frac{1}{K} \sum_{k=1}^{K} \pi_u[c_k] \cdot \gamma(V(d), V(c_k)),$$

where $S(q, d)$ is the non-personalized ranking score, such as that generated by BM25, $\pi_u(c_k)$ is the value for topic c_k in the topic profile of user u, $c_1, \ldots, c_K$ are the top K topics with the largest values $\pi_u(c_k)$, $V(d)$ and $V(c_k)$ are the vector space models of document d and topic c_k, respectively, and $\gamma(\cdot, \cdot)$ is the cosine similarity between two vector space models. Obviously, the more the topic distribution of document d matches with the topic profile of user u, the higher personalized score document d receives. See also [Speretta and Gauch 2005] for other related work.

Qiu and Cho [2006] exploited the first level of ODP and constructed a topic profile π_u for each user u from the documents clicked by the user. They then used the profile in calculating the personalized importance score of page d with respect to user u, given by

$$S(d, u) = \sum_{k=1}^{K} \pi_u[c_k] \times [TSPR_k(d)],$$

where $\mathrm{TSPR}_k(d)$ is the topic-sensitive page rank [Haveliwala 2002] of page d with respect to topic c_k, and $\pi_u[c_k]$ is the score of topic c_k in user u's profile. Intuitively, the more the user u is interested in topic c_k and the higher topic-sensitive page rank score is for page p with respect to topic c_k, the more likely page p is important to the user. Note that $S(d, u)$ represents the importance of web page d and does not depend on a query.

On the one hand, topic-based methods may be applicable to more queries than click-based and term-based methods. The larger coverage is due to the abstraction of user preference at the topic level. On the other hand, topics may be too coarse to represent users' search needs. Consequently, the accuracy of topic-based methods may not be as high as that of click-based and term-based methods [Dou et al. 2007].

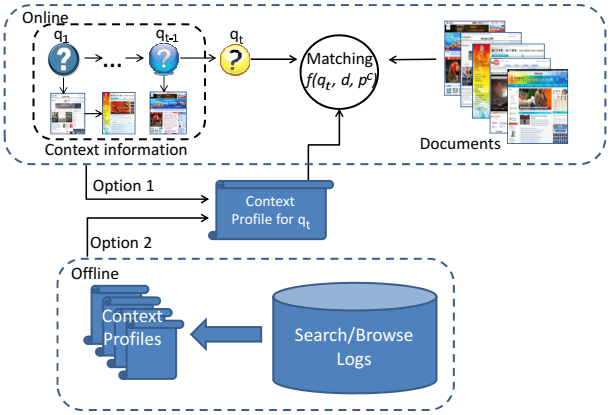

Fig. 9. The general framework of a contextualized search system. A contextualized search system characterizes the search context and leverages the context information in search.

For other topic-based methods, please see [Liu et al. 2002].

5.2 Contextualization

Contextualization is a complementary task to personalization. Instead of building a user profile for each individual user, contextualization creates a context profile to capture the environment of each search activity. Personalization methods take into account an individual user's preference over a long search history. In contrast, contextualization methods take into consideration difference users' preferences in similar short search histories.

Figure 9 shows the process of the contextualization approach. Suppose that a user raises query q_t. A contextualization method characterizes the search context and leverages the context information in search. Different contextualization methods may consider different aspects of the search context, such as the time of the search, the location of the user, the search device from which the query is issued, and the user behavior before raising the current query. In this subsection, we review several studies that focus on user behavior. User behavior is the most intensively studied context information in the state-of-the-art literature. More specifically, given a query q_t, the contextualization methods consider the immediately previous queries, clicked documents, and/or browsed documents that are within the same session of q_t as the *context*.

Given the context information, we may follow two strategies. The first strategy creates a profile directly for the context and integrates the profile into the ranking function. In this case, contextualization is very similar to personalization. The major difference is that contextualization considers a user's short history within a session, while personalization considers a user's long history over several days or even longer time. Therefore, all the personalized techniques discussed in Section 5.1, including the click-based methods, term-based methods, and topic-based methods are applicable here.

For example, Shen et al. [2005a] [2005b] proposed two contextualization methods

using language models. The basic idea is to incorporate the context information of the current query, including the previous queries and the clicked documents, into the query language model. The methods rank the documents high that are not only similar to the current query but also similar to the previous queries and the clicked documents in the same session. The authors evaluated their context-aware model with a small TREC dataset (`http://trec.nist.gov`) and confirmed the effectiveness of their approaches.

Xiang et al. [2010] proposed several context-aware ranking principles for web search, including one based on clicks, two based on terms, and one based on topics. The authors evaluated the ranking principles on large-scale search log data and made two observations. First, at the click level and term level, users are more likely to favor the search results that are different from the contexts. Second, at the topic level, users tend to click on the search results that are consistent with the contexts.

One challenge with the above contextualization strategy is that the context information is usually very sparse. For example, a user typically raises only two or three queries in a session. Moreover, for each query, a user may click on only one or two URLs. Therefore, the context information from a single user may not be sufficient to create an effective context profile. To address this challenge, the second strategy for contextualization creates context profiles by summarizing the common behavior of many users in log data.

Cao et al. [2009] characterized users' search behavior in a session with a variable length Hidden Markov Model (or vlHMM for short) and then conducted contextualization with the vlHMM model. In the vlHMM model, the sequence of queries and clicked URLs in a search session is assumed to be a sequence of observations, and a hidden sequence of states representing topics is assumed to exist behind the sequence of observations. The generative process of the two sequences is determined by the vlHMM model. To learn the vlHMM model, the method first discovers states (topics) by clustering queries and URLs in the click-through bipartite graph using a method by Cao et al. [2008], and then takes the clusters as states. The method then employs the EM algorithm to estimate the parameters of the vlHMM model under the Map-Reduce programming mode [Dean and Ghemawat 2004; 2008]. In contextualization, given a query q_t and its context $O_1, \ldots, O_{t-1}$, the user's search topic can be inferred using the conditional probability distribution of s_t given the query and the context. Once s_t is inferred, the search result can be re-ranked accordingly. Furthermore, query suggestions and URL recommendations based on the context information can also be carried out.

Some other studies related to personalization and contextualization include [Wedig and Madani 2006; Teevan et al. 2008; White et al. 2009; Matthijs and Radlinski 2011; Tyler and Teevan 2010].

6. EVALUATION ON USER SATISFACTION

Another important way of using log data is to evaluate the effectiveness of the search system. In traditional IR, such an evaluation is usually performed on the basis of human labeled data, for example, by taking the Cranfield approach [Cleverdon 1967]. The traditional IR approach faces several challenges when it is applied to

Symbol	Activity
S	A user starts a session
q	A user submits a query
L	A search engine returns a search result page to the user
r	A user clicks on a result
Z	A user exits the session

Table I. A vocabulary of 5 letters representing user activities in search sessions [Fox et al. 2005].

web search. First, the cost of data labeling is high, and the coverage tends to be small. Furthermore, since human judges are not the searchers who submitted the queries, it is often difficult for them to make judgments on the relevance of search results.

6.1 Prediction of User Satisfaction

In recent years, people proposed taking search log data as users' implicit feedback and evaluating the effectiveness of a search system using search log data. By mining and analyzing log data, we can assess how users think about that their information needs are satisfied. In this subsection, we review three representative methods that use search log data to predict users' satisfaction. All the three methods make use of log data in search sessions, but with different models, including sequential pattern model [Fox et al. 2005] and Markov chain [Hassan et al. 2010]. See also [Piwowarski et al. 2009] for other related work.

6.1.1 *Sequential Pattern Model.* Fox et al. [2005] defined a vocabulary of five letters to represent various user activities in search sessions, as shown in Table I. With the vocabulary, a search session can be represented by a sequence of letters. For example, the sequence $SqLrZ$ represents a simple session in which a user starts the session, raises a query, receives a search result page, makes a click on a URL, and leaves the session. The authors then represented all user sessions as sequences of letters and mined frequent patterns from the sequences. To summarize similar patterns, the authors allowed a wild card symbol "*" to match any letter or letter sequence. Finally, the authors investigated the relations between the mined frequent patterns and user satisfaction. The user satisfaction data was collected after the users conducted searches. Some interesting patterns are listed in Table II.

The first behavioral pattern $SqLrZ$ in Table II indicates that a user clicks on a result in the search result page and exits the session immediately. 509 sessions in the data follow this pattern. Among the sessions, 81% of the sessions were explicitly labeled by the users as satisfied, 10% as partially satisfied, and only 7% as dissatisfied. Therefore, pattern $SqlrZ$ has a strong positive correlation with user satisfaction. In other words, a session that follows this pattern likely may make the user satisfied. In contrast, the last behavioral pattern $SqLrLrLrLr*$ indicates that a user keeps browsing search result pages. For this pattern, only 13% of the sessions were labeled as satisfied, 35% as partially satisfied, and 51% as dissatisfied. The pattern has a strong negative correlation with user satisfaction.

6.1.2 *Markov Chain Models.* Hassan et al. [2010] and Hassan [2012] extended the vocabulary of search activities defined by Fox et al. [2005], and constructed a

Pattern	Freq.	%SAT	%PSAT	%DSAT
SqLrZ	509	81	10	7
SqLrLZ	117	75	15	9
SqLrLrZ	82	73	13	13
SqLrqLr*	70	64	25	10
SqLrLrLrZ	61	57	22	19
SqLrLr*	362	23	39	36
SqLrLrLr*	129	20	37	42
SqLrLrLrLr*	114	13	35	51

Table II. The correlation between sequential user behavior patterns in sessions and user satisfaction [Fox et al. 2005]. For example, pattern $SqlrZ$ has a strong positive correlation with user satisfaction, while pattern $SqLrLrLrLr*$ has a strong negative correlation with user satisfaction.

Markov chain to model user behavior in search sessions. They used a vocabulary $V = \{Q, SR, AD, RL, SP, SC, OTH\}$, where Q represents submission of a query by a user, SR represents a click on a search result, AD represents a click on an advertisement, RL represents a click on a query suggestion, SP represents a click on a spelling suggestion, SC represents a click on a deep link, and OTH represents any other click. They then took search sessions as sequences of activities and constructed a Markov chain on the sequences. The Markov chain is defined as a tuple (V, E, w), where V is the vocabulary of letters denoting user activities, E is the set of transition edges between activities, and w is the set of transition probabilities associated with the edges. In training, the transition probability from activity s_i to actitivity s_j is estimated from log data using maximum Likelihood estimation $P(s_i, s_j) = \frac{N_{s_i,s_j}}{N_{s_i}}$, where N_{s_i,s_j} is the number of times the transition from s_i to s_j is observed in the log data, and N_{s_i} is the total number of times s_i is observed.

They assumed that user satisfactions on search sessions can be labeled by human judges. Specifically, a judge can go through a user session and infer whether the user's information need is fulfilled. The judge can then label the session as successful or not. The authors then learned two Markov chains, denoted by M_s and M_f, respectively, from all the sessions labeled as successful and unsuccessful. Given a user session $S = s_1 \dots s_k$, where $s_i \in V$ $(1 \leq i \leq k)$, the likelihood of S with respect to the successful Markov chain LL_{M_s} and the likelihood $LL_{M_f}(S)$ of S with respect to the unsuccessful model M_f can be calculated. Finally, the session is predicted as successful if the ratio of LL_{M_s} over LL_{M_f} is greater than a threshold τ, and unsuccessful otherwise.

The authors further incorporated time features into the Markov model. The basic idea is that the distribution of transition time between activities can be very different in successful and unsuccessful sessions. For example, Figure 10 shows the distribution of time for the transition from activity SR to activity Q for successful sessions and unsuccessful sessions. The two curves have very different shapes and thus can be leveraged to differentiate successful and unsuccessful sessions. The authors assumed that the transition time follows the Gamma distribution and estimated the parameters from successful and unsuccessful sessions, respectively. Finally, the time distribution can be either used independently or integrated into the

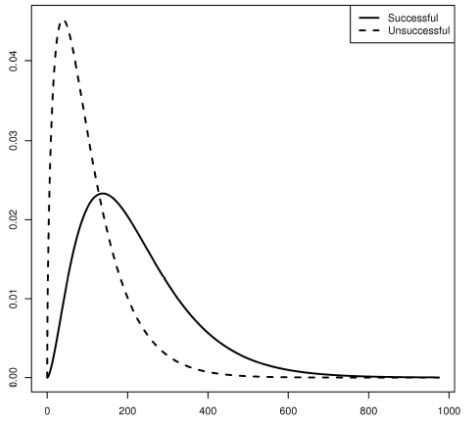

Fig. 10. Time distributions of $SR \rightarrow Q$ transitions for successful and unsuccessful search sessions, where Q represents submission of a query and SR represents a click on a search result (extracted from [Hassan et al. 2010]). The distributions of successful and unsuccessful sessions have different shapes.

transition probabilities to make predictions on the success of sessions.

6.1.3 *Discussion.* The studies discussed in this section have three major differences from those in Section 4.

(1) The purpose of the models in Section 4 is to predict the relevance of documents with respect to a single query and help to rank or re-rank documents according to their relevance. In contrast, the purpose of the models in this section is to predict user satisfaction.
(2) Most of the models in Section 4 focus on the user behavior with respect to a single query. In contrast, the models in this section consider multiple queries in a search session. It is reasonable to consider a longer context of search to make more accurate prediction of user satisfaction.
(3) The models in Section 4 mainly consider users' click-through information. In contrast, most of the models in this section go beyond the use of click-through information.

7. SUMMARY

In this paper, we presented a survey on search and browse log mining for web search, with the focus on improving the effectiveness of web search by query understanding, document understanding, document ranking, user understanding, and monitoring and feedbacks. As reviewed, many advanced techniques were developed. Those techniques were applied to huge amounts of search and browse log data available at web search engines, and were powerful in enhancing the quality of the search engines.

There are still many challenging and interesting problems for future work. We list three of them here as examples.

First, it is challenging to deal with the long tail in search and browsing log

effectively. Search and browse log data are user behavior data and follow the power law distributions in many aspects. Usually it is easy to mine useful knowledge from the head part of a power law distribution (for example, [Spink et al. 2002]). How to propagate the mined knowledge from the head part to the tail part is still a challenge for most of the log mining tasks.

Second, it is important to leverage other information or knowledge in mining. Log mining mainly focuses on the use of log data. It would be helpful to leverage information or knowledge in other data sources during the mining process, such as Wikipedia. It is necessary to conduct more research on log mining in such a setting.

Last, privacy preserving log mining remains a grand challenge. In 2006, AOL released a search log dataset, but unfortunately a privacy issue arose in the data release. The identity of a user can be detected from the data, although certain data processing had been done in advance [Barbaro and Zeller 2006]. How to preserve privacy in log data and in the meantime do not sacrifice the utility of the log data is a critical research issue.

REFERENCES

Agichtein, E. 2010. Inferring searcher intent. In *WWW'10 Tutorial*.

Agichtein, E., Brill, E., and Dumais, S. 2006. Improving web search ranking by incorporating user behavior information. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '06. ACM, New York, NY, USA, 19–26.

Agichtein, E., Brill, E., Dumais, S., and Ragno, R. 2006. Learning user interaction models for predicting web search result preferences. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '06. ACM, New York, NY, USA, 3–10.

Backstrom, L., Kleinberg, J., Kumar, R., and Novak, J. 2008. Spatial variation in search engine queries. In *Proceedings of the 17th International Conference on World Wide Web*. WWW '08. ACM, New York, NY, USA, 357–366.

Baeza-Yates, R. 2004. Web mining in search engines.

Baeza-Yates, R. A., Calderón-Benavides, L., and González-Caro, C. N. 2006. The intention behind web queries. In *Proceedings of the 13th International Conference on String Processing and Information Retrieval*. 98–109.

Baeza-Yates, R. A. and Ribeiro-Neto, B. 1999. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Barbaro, M. and Zeller, T. J. 2006. A face is exposed for aol searcher no. 4417749. *The New York Times*.

Beeferman, D. and Berger, A. 2000. Agglomerative clustering of a search engine query log. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '00. ACM, New York, NY, USA, 407–416.

Beitzel, S. M., Jensen, E. C., Chowdhury, A., Frieder, O., and Grossman, D. 2007. Temporal analysis of a very large topically categorized web query log. *Journal of American Society of Information Science and Technology 58*, 166–178.

Beitzel, S. M., Jensen, E. C., Chowdhury, A., Grossman, D., and Frieder, O. 2004. Hourly analysis of a very large topically categorized web query log. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '04. ACM, New York, NY, USA, 321–328.

Beitzel, S. M., Jensen, E. C., Lewis, D. D., Chowdhury, A., and Frieder, O. 2007. Automatic classification of web queries using very large unlabeled query logs. *ACM Transactions on Information Systems 25,* 2.

Bendersky, M. and Croft, W. B. 2009. Analysis of long queries in a large scale search log. In *Proceedings of the 2009 Workshop on Web Search Click Data*. WSCD '09. ACM, New York, NY, USA, 8–14.

Bennett, P. N., White, R. W., Chu, W., Dumais, S. T., Bailey, P., Borisyuk, F., and Cui, X. 2012. Modeling the impact of short- and long-term behavior on search personalization. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '12. ACM, New York, NY, USA, 185–194.

Bergsma, S. and Wang, Q. I. 2007. Learning noun phrase query segmentation. In *Proceedings of Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. 819–826.

Berry, M. W. 2003. *Survey of Text Mining*. Springer-Verlag New York, Inc.

Boldi, P., Bonchi, F., Castillo, C., Donato, D., Gionis, A., and Vigna, S. 2008. The query-flow graph: Model and applications. In *Proceedings of the 17th ACM conference on Information and Knowledge Management*. CIKM '08. ACM, New York, NY, USA, 609–618.

Broder, A. 2002. A taxonomy of web search. *SIGIR Forum 36*, 3–10.

Cao, H., Jiang, D., Pei, J., Chen, E., and Li, H. 2009. Towards context-aware search by learning a very large variable length hidden markov model from search logs. In *Proceedings of the 18th International Conference on World Wide Web*. WWW '09. ACM, New York, NY, USA, 191–200.

Cao, H., Jiang, D., Pei, J., He, Q., Liao, Z., Chen, E., and Li, H. 2008. Context-aware query suggestion by mining click-through and session data. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '08. ACM, New York, NY, USA, 875–883.

Chapelle, O. and Zhang, Y. 2009. A dynamic bayesian network click model for web search ranking. In *Proceedings of the 18th International Conference on World Wide Web*. WWW '09. ACM, New York, NY, USA, 1–10.

Chien, S. and Immorlica, N. 2005. Semantic similarity between search engine queries using temporal correlation. In *Proceedings of the 14th International Conference on World Wide Web*. WWW '05. ACM, New York, NY, USA, 2–11.

Cleverdon, C. 1967. The cranfield tests on index language devices. *Aslib Proceedings 19,* 173-192.

Craswell, N. and Szummer, M. 2007. Random walks on the click graph. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '07. ACM, New York, NY, USA, 239–246.

Craswell, N., Zoeter, O., Taylor, M., and Ramsey, B. 2008. An experimental comparison of click position-bias models. In *Proceedings of the International Conference on Web search and web Data Mining*. WSDM '08. ACM, New York, NY, USA, 87–94.

Croft, W. B., Bendersky, M., Li, H., and Xu, G. 2010. Query representation and understanding workshop. *SIGIR Forum 44,* 2, 48–53.

Croft, W. B., Metzler, D., and Strohman, T. 2009. *Search Engines - Information Retrieval in Practice*. Pearson Education.

Cui, H., Wen, J.-R., Nie, J.-Y., and Ma, W.-Y. 2002. Probabilistic query expansion using query logs. In *Proceedings of the Eleventh International Conference on World Wide Web*. WWW '02. ACM, New York, NY, USA, 325–332.

Dean, J. and Ghemawat, S. 2004. Mapreduce: Simplified data processing on large clusters. In *Proceedings of the 6th Symposium on Operating Systems Design & Implementation*. USENIX Association, Berkeley, CA, USA, 10–10.

Dean, J. and Ghemawat, S. 2008. Mapreduce: Simplified data processing on large clusters. *Communications of ACM 51*, 107–113.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science 41*, 391–407.

Dong, A., Chang, Y., Zheng, Z., Mishne, G., Bai, J., Zhang, R., Buchner, K., Liao, C., and Diaz, F. 2010. Towards recency ranking in web search. In *Proceedings of the Third ACM*

International Conference on Web search and Data Mining. WSDM '10. ACM, New York, NY, USA, 11–20.

Dou, Z., Song, R., and Wen, J.-R. 2007. A large-scale evaluation and analysis of personalized search strategies. In *Proceedings of the 16th International Conference on World Wide Web*. WWW '07. ACM, New York, NY, USA, 581–590.

Dou, Z., Song, R., Yuan, X., and Wen, J.-R. 2008. Are click-through data adequate for learning web search rankings? In *Proceedings of the 17th ACM conference on Information and Knowledge Management*. CIKM '08. ACM, New York, NY, USA, 73–82.

Duan, H. and Hsu, B.-J. P. 2011. Online spelling correction for query completion. In *Proceedings of the 20th International Conference on World Wide Web*. WWW '11. ACM, New York, NY, USA, 117–126.

Dupret, G. E. and Piwowarski, B. 2008. A user browsing model to predict search engine click data from past observations. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '08. ACM, New York, NY, USA, 331–338.

Fonseca, B. M., Golgher, P., Pôssas, B., Ribeiro-Neto, B., and Ziviani, N. 2005. Concept-based interactive query expansion. In *Proceedings of the Fourteenth ACM International Conference on Information and Knowledge Management*. CIKM '05. ACM, New York, NY, USA, 696–703.

Fox, S., Karnawat, K., Mydland, M., Dumais, S., and White, T. 2005. Evaluating implicit measures to improve web search. *ACM Transactions on Information Systems 23*, 147–168.

Fuxman, A., Tsaparas, P., Achan, K., and Agrawal, R. 2008. Using the wisdom of the crowds for keyword generation. In *Proceedings of the 17th International Conference on World Wide Web*. WWW '08. ACM, New York, NY, USA, 61–70.

Gao, J., Yuan, W., Li, X., Deng, K., and Nie, J.-Y. 2009. Smoothing clickthrough data for web search ranking. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '09. ACM, New York, NY, USA, 355–362.

Guo, F., Liu, C., Kannan, A., Minka, T., Taylor, M. J., Wang, Y. M., and Faloutsos, C. 2009. Click chain model in web search. In *Proceedings of 18th International Conference on World Wide Web*. 11–20.

Guo, F., Liu, C., and Wang, Y. M. 2009. Efficient multiple-click models in web search. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*. WSDM '09. ACM, New York, NY, USA, 124–131.

Guo, J., Xu, G., Li, H., and Cheng, X. 2008. A unified and discriminative model for query refinement. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '08. ACM, New York, NY, USA, 379–386.

Hagen, M., Potthast, M., Beyer, A., and Stein, B. 2012. Towards optimum query segmentation: in doubt without. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*. 1015–1024.

Hagen, M., Potthast, M., Stein, B., and Bräutigam, C. 2011. Query segmentation revisited. In *Proceedings of the 20th International Conference on World Wide Web*. 97–106.

Hassan, A. 2012. A semi-supervised approach to modeling web search satisfaction. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '12. ACM, New York, NY, USA, 275–284.

Hassan, A., Jones, R., and Klinkner, K. L. 2010. Beyond dcg: User behavior as a predictor of a successful search. In *Proceedings of the Third ACM International Conference on Web search and Data Mining*. WSDM '10. ACM, New York, NY, USA, 221–230.

Haveliwala, T. H. 2002. Topic-sensitive pagerank. In *Proceedings of the 11th International Conference on World Wide Web*. WWW '02. ACM, New York, NY, USA, 517–526.

He, D., Göker, A., and Harper, D. J. 2002. Combining evidence for automatic web session identification. *Information Processing and Management 38*, 727–742.

Hölscher, C. and Strube, G. 2000. Web search behavior of internet experts and newbies. In *Proceedings of the 9th International World Wide Web Conference on Computer Networks*. North-Holland Publishing Co., Amsterdam, The Netherlands, The Netherlands, 337–346.

Hu, B., Zhang, Y., Chen, W., Wang, G., and Yang, Q. 2011. Characterizing search intent diversity into click models. In *Proceedings of the 20th International Conference on World Wide Web*. WWW '11. ACM, New York, NY, USA, 17–26.

Hu, J., Wang, G., Lochovsky, F., Sun, J.-t., and Chen, Z. 2009. Understanding user's query intent with wikipedia. In *Proceedings of the 18th International Conference on World Wide Web*. WWW '09. ACM, New York, NY, USA, 471–480.

Hu, Y., Qian, Y., Li, H., Jiang, D., Pei, J., and Zheng, Q. 2012. Mining query subtopics from search log data. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '12. ACM, New York, NY, USA, 305–314.

Huang, C.-K., Chien, L.-F., and Oyang, Y.-J. 2003. Relevant term suggestion in interactive web search based on contextual information in query session logs. *Journal of American Society of Information Science and Technology 54,* 7 (May), 638–649.

Jansen, B. J. and Pooch, U. W. 2001. A review of web searching studies and a framework for future research. *Journal of the American Society for Information Science and Technology 52,* 3, 235–246.

Jansen, B. J. and Spink, A. 2006. How are we searching the world wide web?: A comparison of nine search engine transaction logs. *Information Processing and Management 42*, 248–263.

Jansen, B. J., Spink, A., Blakely, C., and Koshman, S. 2007. Defining a session on web search engines: Research articles. *Journal of American Society of Information Science and Technology 58*, 862–871.

Ji, M., Yan, J., Gu, S., Han, J., He, X., Zhang, W. V., and Chen, Z. 2011. Learning search tasks in queries and web pages via graph regularization. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '11. ACM, New York, NY, USA, 55–64.

Jin, X., Zhou, Y., and Mobasher, B. 2004. Web usage mining based on probabilistic latent semantic analysis. In *Proceedings of the tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '04. ACM, New York, NY, USA, 197–205.

Joachims, T. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '02. ACM, New York, NY, USA, 133–142.

Joachims, T., Granka, L., Pan, B., Hembrooke, H., and Gay, G. 2005. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '05. ACM, New York, NY, USA, 154–161.

Jones, K. S., Walker, S., and Robertson, S. 1998. Probabilistic model of information retrieval: Development and status. Tech. Rep. TR-446, Cambridge University Computer Laboratory.

Jones, R. and Klinkner, K. L. 2008. Beyond the session timeout: Automatic hierarchical segmentation of search topics in query logs. In *Proceedings of the 17th ACM conference on Information and Knowledge Management*. CIKM '08. ACM, New York, NY, USA, 699–708.

Jones, R., Rey, B., Madani, O., and Greiner, W. 2006. Generating query substitutions. In *Proceedings of the Fifteenth International Conference on World Wide Web*. WWW '06. ACM, New York, NY, USA, 387–396.

Kang, D., Jiang, D., Pei, J., Liao, Z., Sun, X., and Choi, H.-J. 2011. Multidimensional mining of large-scale search logs: A topic-concept cube approach. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*. 385–394.

Kleinberg, J. M. 1999. Authoritative sources in a hyperlinked environment. *Journal of ACM 46*, 604–632.

Kulkarni, A., Teevan, J., Svore, K. M., and Dumais, S. T. 2011. Understanding temporal query dynamics. In *Proceedings of the Fourth ACM International Conference on Web search and Data Mining*. WSDM '11. ACM, New York, NY, USA, 167–176.

Lafferty, J. and Zhai, C. 2001. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '01. ACM, New York, NY, USA, 111–119.

Lathauwer, L. D., Moor, B. D., and Vandewalle, J. 2000. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications 21,* 4 (Mar.), 1253–1278.

Lee, U., Liu, Z., and Cho, J. 2005. Automatic identification of user goals in web search. In *Proceedings of the 14th International Conference on World Wide Web*. WWW '05. ACM, New York, NY, USA, 391–400.

Li, H. 2011. Learning to rank for information retrieval and natural language processing. *Synthesis Lectures on Human Language Technologies 4,* 1, 1–113.

Li, X., Wang, Y.-Y., and Acero, A. 2008. Learning query intent from regularized click graphs. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '08. ACM, New York, NY, USA, 339–346.

Li, Y., Duan, H., and Zhai, C. 2012. A generalized hidden markov model with discriminative training for query spelling correction. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '12. ACM, New York, NY, USA, 611–620.

Li, Y., Hsu, B.-J. P., Zhai, C., and Wang, K. 2011. Unsupervised query segmentation using clickthrough for information retrieval. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '11. ACM, New York, NY, USA, 285–294.

Liao, Z., Jiang, D., Chen, E., Pei, J., Cao, H., and Li, H. 2011. Mining concept sequences from large-scale search logs for context-aware query suggestion. *ACM Transactions on Intelligent System and Technology 3,* 1, 17.

Liao, Z., Song, Y., He, L.-w., and Huang, Y. 2012. Evaluating the effectiveness of search task trails. In *Proceedings of the 21st International Conference on World Wide Web*. WWW '12. ACM, New York, NY, USA, 489–498.

Liu, C., Guo, F., and Faloutsos, C. 2009. Bbm: Bayesian browsing model from petabyte-scale data. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '09. ACM, New York, NY, USA, 537–546.

Liu, F., Yu, C., and Meng, W. 2002. Personalized web search by mapping user queries to categories. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management*. CIKM '02. ACM, New York, NY, USA, 558–565.

Liu, Y., Gao, B., Liu, T.-Y., Zhang, Y., Ma, Z., He, S., and Li, H. 2008. Browserank: Letting web users vote for page importance. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '08. ACM, New York, NY, USA, 451–458.

Lucchese, C., Orlando, S., Perego, R., Silvestri, F., and Tolomei, G. 2011. Identifying task-based sessions in search engine query logs. In *Proceedings of the Fourth ACM International Conference on Web search and Data Mining*. WSDM '11. ACM, New York, NY, USA, 277–286.

Matthijs, N. and Radlinski, F. 2011. Personalizing web search using long term browsing history. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*. 25–34.

Mei, Q. and Church, K. 2008. Entropy of search logs: how hard is search? with personalization? with backoff? In *Proceedings of the International Conference on Web search and web Data Mining*. WSDM '08. ACM, New York, NY, USA, 45–54.

Mei, Q., Zhou, D., and Church, K. 2008. Query suggestion using hitting time. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*. CIKM '08. ACM, New York, NY, USA, 469–478.

Ozertem, U., Chapelle, O., Donmez, P., and Velipasaoglu, E. 2012. Learning to suggest: A machine learning framework for ranking query suggestions. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '12. ACM, New York, NY, USA, 25–34.

Paşca, M. 2007. Organizing and searching the world wide web of facts – step two: Harnessing the wisdom of the crowds. In *Proceedings of the 16th International Conference on World Wide Web*. WWW '07. ACM, New York, NY, USA, 101–110.

Paşca, M. and Alfonseca, E. 2009. Web-derived resources for web information retrieval: from conceptual hierarchies to attribute hierarchies. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '09. ACM, New York, NY, USA, 596–603.

Page, L., Brin, S., Motwani, R., and Winograd, T. 1999. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab. November. Previous number = SIDL-WP-1999-0120.

Pitkow, J., Schütze, H., Cass, T., Cooley, R., Turnbull, D., Edmonds, A., Adar, E., and Breuel, T. 2002. Personalized search. *Communications of ACM 45,* 9, 50–55.

Piwowarski, B., Dupret, G., and Jones, R. 2009. Mining user web search activity with layered bayesian networks or how to capture a click in its context. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*. WSDM '09. ACM, New York, NY, USA, 162–171.

Poblete, B. and Baeza-Yates, R. 2008. Query-sets: Using implicit feedback and query patterns to organize web documents. In *Proceedings of the 17th International Conference on World Wide Web*. WWW '08. ACM, New York, NY, USA, 41–50.

Pretschner, A. and Gauch, S. 1999. Ontology based personalized search. In *Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence*. ICTAI '99. IEEE Computer Society, Washington, DC, USA, 391–.

Qiu, F. and Cho, J. 2006. Automatic identification of user interest for personalized search. In *Proceedings of the 15th International Conference on World Wide Web*. WWW '06. ACM, New York, NY, USA, 727–736.

Radlinski, F. and Joachims, T. 2005. Query chains: Learning to rank from implicit feedback. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*. KDD '05. ACM, New York, NY, USA, 239–248.

Radlinski, F. and Joachims, T. 2006. Minimally invasive randomization for collecting unbiased preferences from clickthrough logs. In *Proceedings of the 21st National Conference on Artificial Intelligence*. AAAI Press, 1406–1412.

Radlinski, F. and Joachims, T. 2007. Active exploration for learning rankings from clickthrough data. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '07. ACM, New York, NY, USA, 570–579.

Rose, D. E. and Levinson, D. 2004. Understanding user goals in web search. In *Proceedings of the 13th International Conference on World Wide Web*. WWW '04. ACM, New York, NY, USA, 13–19.

Salton, G., Wong, A., and Yang, C. S. 1975. A vector space model for automatic indexing. *Communications of ACM 18*, 613–620.

Shen, D., Pan, R., Sun, J.-T., Pan, J. J., Wu, K., Yin, J., and Yang, Q. 2006. Query enrichment for web-query classification. *ACM Transactions on Information Systems 24,* 3, 320–352.

Shen, D., Sun, J.-T., Yang, Q., and Chen, Z. 2006. Building bridges for web query classification. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '06. ACM, New York, NY, USA, 131–138.

Shen, X., Tan, B., and Zhai, C. 2005a. Context-sensitive information retrieval using implicit feedback. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '05. ACM, New York, NY, USA, 43–50.

Shen, X., Tan, B., and Zhai, C. 2005b. Implicit user modeling for personalized search. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*. CIKM '05. ACM, New York, NY, USA, 824–831.

Shokouhi, M. and Radinsky, K. 2012. Time-sensitive query auto-completion. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '12. ACM, New York, NY, USA, 601–610.

Silverstein, C., Marais, H., Henzinger, M., and Moricz, M. 1999. Analysis of a very large web search engine query log. *SIGIR Forum 33*, 6–12.

Silvestri, F. 2010. Mining query logs: Turning search usage data into knowledge. *Foundations and Trends in Information Retrieval 4,* 1-2, 1–174.

Speretta, M. and Gauch, S. 2005. Personalized search based on user search histories. In *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence*. WI '05. IEEE Computer Society, Washington, DC, USA, 622–628.

Spink, A., Jansen, B. J., Wolfram, D., and Saracevic, T. 2002. From e-sex to e-commerce: Web search changes. *Computer 35*, 107–109.

Spink, A., Wolfram, D., Jansen, M. B. J., and Saracevic, T. 2001. Searching the web: the public and their queries. *Journal of American Society of Information Science and Technology 52,* 3 (Feb.), 226–234.

Sun, J.-T., Zeng, H.-J., Liu, H., Lu, Y., and Chen, Z. 2005. Cubesvd: A novel approach to personalized web search. In *Proceedings of the 14th international conference on World Wide Web*. 382–390.

Szpektor, I., Gionis, A., and Maarek, Y. 2011. Improving recommendation for long-tail queries via templates. In *Proceedings of the 20th International Conference on World Wide Web*. WWW '11. ACM, New York, NY, USA, 47–56.

Tan, B., Shen, X., and Zhai, C. 2006. Mining long-term search history to improve search accuracy. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '06. ACM, New York, NY, USA, 718–723.

Teevan, J., Adar, E., Jones, R., and Potts, M. A. S. 2007. Information re-retrieval: Repeat queries in yahoo's logs. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '07. ACM, New York, NY, USA, 151–158.

Teevan, J., Dumais, S. T., and Horvitz, E. 2005. Personalizing search via automated analysis of interests and activities. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '05. ACM, New York, NY, USA, 449–456.

Teevan, J., Dumais, S. T., and Liebling, D. J. 2008. To personalize or not to personalize: Modeling queries with variation in user intent. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '08. ACM, New York, NY, USA, 163–170.

Tyler, S. K. and Teevan, J. 2010. Large scale query log analysis of re-finding. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*. 191–200.

Vlachos, M., Meek, C., Vagena, Z., and Gunopulos, D. 2004. Identifying similarities, periodicities and bursts for online search queries. In *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data*. SIGMOD '04. ACM, New York, NY, USA, 131–142.

Wang, K., Gloy, N., and Li, X. 2010. Inferring search behaviors using partially observable markov (pom) model. In *Proceedings of the Third ACM International Conference on Web search and Data Mining*. WSDM '10. ACM, New York, NY, USA, 211–220.

Weber, I. and Jaimes, A. 2011. Who uses web search for what? and how? In *Proceedings of the Fourth ACM International Conference on Web search and Data Mining*. WSDM '11. ACM, New York, NY, USA, 15–24.

Wedig, S. and Madani, O. 2006. A large-scale analysis of query logs for assessing personalization opportunities. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '06. ACM, New York, NY, USA, 742–747.

Weerkamp, W., Berendsen, R., Kovachev, B., Meij, E., Balog, K., and de Rijke, M. 2011. People searching for people: Analysis of a people search engine log. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '11. ACM, New York, NY, USA, 45–54.

Welch, M. J. and Cho, J. 2008. Automatically identifying localizable queries. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '08. ACM, New York, NY, USA, 507–514.

Wen, J.-R., Nie, J.-Y., and Zhang, H.-J. 2001. Clustering user queries of a search engine. In *Proceedings of the Tenth International Conference on World Wide Web*. WWW '01. ACM, New York, NY, USA, 162–168.

White, R. W., Bailey, P., and Chen, L. 2009. Predicting user interests from contextual information. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '09. ACM, New York, NY, USA, 363–370.

Wolfram, D., Spink, A., Jansen, B. J., and Saracevic, T. 2001. Vox populi: The public searching of the web. *Journal of the American Society for Information Science and Technology 52,* 12, 1073–1074.

Xiang, B., Jiang, D., Pei, J., Sun, X., Chen, E., and Li, H. 2010. Context-aware ranking in web search. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '10. ACM, New York, NY, USA, 451–458.

Xu, J. and Xu, G. 2011. Learning similarity function for rare queries. In *Proceedings of the Fourth ACM International Conference on Web search and Data Mining*. WSDM '11. ACM, New York, NY, USA, 615–624.

Xue, G.-R., Zeng, H.-J., Chen, Z., Yu, Y., Ma, W.-Y., Xi, W., and Fan, W. 2004. Optimizing web search using web click-through data. In *Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management*. CIKM '04. ACM, New York, NY, USA, 118–126.

Yi, X., Raghavan, H., and Leggetter, C. 2009. Discovering users' specific geo intention in web search. In *Proceedings of the 18th International Conference on World Wide Web*. WWW '09. ACM, New York, NY, USA, 481–490.

Zhu, G. and Mishne, G. 2009. Mining rich session context to improve web search. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '09. ACM, New York, NY, USA, 1037–1046.

...