Analysis of Instant Search Query Logs *

Inci Cetindil
University of California, Irvine
Irvine, California 92697, USA
icetindi@ics.uci.edu

Jamshid Esmaelnezhad
University of California, Irvine
Irvine, California 92697, USA
jesmaeln@ics.uci.edu

Chen Li
University of California, Irvine
Irvine, California 92697, USA
chenli@ics.uci.edu

David Newman
University of California, Irvine
Irvine, California 92697, USA
newman@uci.edu

ABSTRACT

Instant search is a new search paradigm that shows results as a user types in a query. It has become increasingly popular in recent years due to its simplicity and power. In an instant-search system, every keystroke from a user triggers a new request to the server. Therefore, its log has a richer content than that of a traditional search system, and previous log analysis research is not applicable to this type of log. In this paper, we study the problem of analyzing the query log of an instant-search system. We propose a classification scheme for user typing behaviors. We also compare the log of an instant-search system and that of a traditional search system on the same data. The results show that on a people directory search system, instant search can typically save 2 seconds per search, reduce the typing effort by showing the results with fewer characters entered, and increase the success rate.

Categories and Subject Descriptors

H.1.2 [**Information Systems**]: User/Machine Systems—*human information processing*; H.3.3 [**Information Systems**]: Information Search and Retrieval—*query formulation, search process*

General Terms

Experimentation, Human Factors

Keywords

instant search, log analysis, user behavior

1. INTRODUCTION

*A full version of this paper is available as *Analysis of Instant Search Query Logs* at `http://ipubmed.ics.uci.edu/pubs/`

* Inci Cetindil and Chen Li have financial interest in Bimaple Technology Inc., a company currently commercializing some of the techniques described in this publication. This work is partially supported by the NIH grant 1R21LM010143-01A1.

Fifteenth International Workshop on the Web and Databases (WebDB 2012),
May 20, 2012 - Scottsdale, AZ, USA.

Increasing Popularity of Instant Search: The goal of information systems is to allow users to find results to their queries, and do so quickly. Keyword search is a widely accepted method for achieving this goal due to its simplicity, and has been used to search document collections and relational data. In traditional keyword search, a user composes a complete query and submits it to the system to find relevant search results. This search paradigm requires the user to formulate a correctly spelled, complete query to find relevant results. When the user has limited knowledge about the data being searched, they can feel "left in the dark" when issuing queries, and have to resort to a try-and-see approach for modifying the query to find relevant results.

To address this problem, many search interfaces provide instant feedback as users formulate search queries. Most search engines and many online search forms support instant search, which shows suggested queries or results on the fly as a user types in a query character by character. These instant-search systems can be classified into two categories: *instant-suggestion systems* and *instant-result systems*. An instant-suggestion system continuously suggests relevant queries. The PubMed[1] service is such a system (as of February 2012). For example, if a user types in "`hom`", the system first predicts several possible queries, such as "`homologous`" and "`home care`". Then, the user may choose one of the suggestions and submit that query to the search engine to retrieve the results. Most instant-suggestion systems rely on query logs to extract frequent queries. Instant suggestion is also possible without query logs [3, 7]. On the other hand, an instant-result system shows the search results as a user types. For instance, the search interface at Netflix[2] allows a user to search for movies by their titles, actors, directors, or genres (as of February 2012). If a user types in a partial query, the system shows a movie matching this keyword as a prefix. For example, after "`bat`" is typed, "`Batman Forever`" and "`Battlestar Galactica`" are displayed.

The problem: In this paper we study a problem relevant to such systems: *how to analyze the query log of an instant-search system?* In particular, we want to answer the following two questions: (1) *What are the user behaviors in instant search?* (2) *What are the quantifiable benefits of instant search?*

The first question is about how user behaviors are affected by this new type of search system. Understanding

[1] `http://www.ncbi.nlm.nih.gov/pubmed/`
[2] `http://www.netflix.com/BrowseSelection`

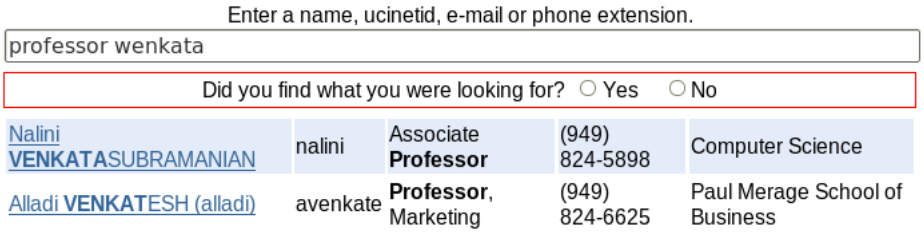


Figure 1: Instant-fuzzy search on the UC Irvine people directory (http://psearch.ics.uci.edu).

the user behaviors of a search system can help improve the search experience. The interaction between users and an instant-search system is different from that of a traditional search system. Users are guided by the instant feedback as they are typing their queries, which reduces the need for trial-and-error searching. Users benefit from the continuous feedback not only in formulating the query, but also in understanding the underlying data. Seeing the results on the fly also reduces typing effort, since relevant results are often displayed before users complete their typing. For example, a screenshot of instant search on a people directory is shown in Figure 1, where the user typed in "`wenkata`" and found a long last name "`venkatasubramanian`". In a traditional search system the expectation of most users is that they need to type the complete keywords before submitting the query. Usually, instant-search users find their results faster. For example, Google Instant, one of the most popular instant-search engines, estimates that it can save 2-5 seconds per search [1]. These changes in the interaction between the user and the search system indicate the influence of an instant-search system on the user's decision process.

The content of the query log in an instant-search system differs from that of a traditional search engine. A request is sent to an instant-search system for each keystroke. Therefore, the log contains more detailed information than that of traditional systems about users' actions. These instant search logs also reveal exactly how users typed their queries. We aim to explore this unique feature of an instant-search log to gain more insight about user behaviors.

The second question we want to answer is how to quantify the benefits of instant search over traditional search. As described in Section 2, we have both an instant-search system and a traditional search system over the same university people directory. Both systems have been frequently used by students, staff, and faculty for over four years. By analyzing their query logs, we can make an "apples-to-apples" comparison, and measure the benefits of instant search over traditional search.

Contributions: In this paper, we study the problem of analyzing instant-search logs and provide answers to these questions. We analyze query logs of an instant-search system to understand user behaviors. As a result of this analysis we show the benefits of instant search. More specifically, we make the following contributions. (1) We analyze the query log of an instant-search system, and classify sessions based on different user typing behaviors. Then we present statistics obtained from the log (Section 3). (2) We show the benefits of instant search compared to traditional search in terms of user effort, time required to fulfill an information need, and success rate. To make a fair comparison, we propose methods to estimate the statistics of missing information in a traditional search log. We conduct a user study to compare the user satisfaction on these search systems. The comparison showed that instant search can typically save 2 seconds and can also increase the success rate of a search (Section 4). To the best of our knowledge, our work is the first study analyzing instant-search query logs.

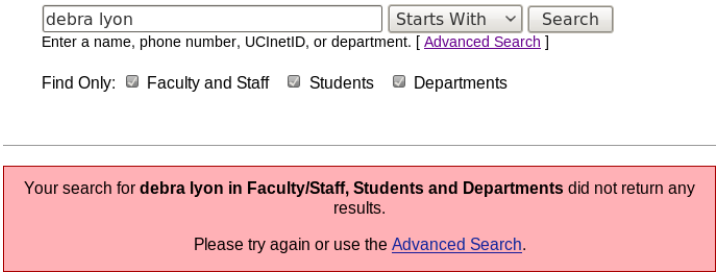
1.1 Related Work

Query Log Analysis: There have been many studies in analyzing the query logs of Web search engines [4, 10, 11]. These studies generally focused on measurements such as the average query length and session length, and query classification based on their topics. Recent studies on mobile phone query log analysis show that the search patterns on smart phones resemble the patterns on computers more than the patterns on mobile phones [2, 5]. Most of the research on log analysis is about Web search engines. There are an increasing number of vertical search engines for specific domains. LinkedIn[3] is an example, which is a professional networking site specialized in people search. Weerkamp et al. [12] investigated whether user behaviors on vertical search engines differ from that of Web search engines. They conducted an analysis on the query log of a commercial people search engine and reported smaller average query length than Web search engines. In this paper, we analyze the query log of a vertical people search engine as described in Section 2. Since its instant-search feature changes the way users formulate queries, its query log needs to be analyzed using a different methodology from traditional query log analysis methods.

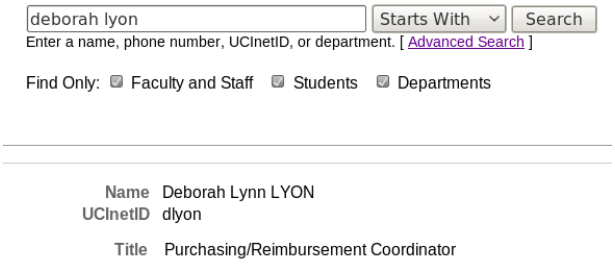
Session Boundary Detection: One challenge in query log analysis is to detect session boundaries. Sessions are needed to separate each information need of a user. Silverstein et al. [10] defined a session as a series of queries by a single user made within a small range of time. They restricted each session to a small range of time based on the intuition that a user fulfills a single information need without a major interruption, and used a 5-minute cutoff value. As long as the time difference between two consecutive queries was smaller than 5 minutes, these two queries were considered to belong to the same session. For detecting session boundaries, other than using time-based session separation, Ozmutlu and Çavdur [9] used a genetic algorithm to automatically identify the different topics in a sequence of queries issued by the same user. In this study, we focus on the similarity of consecutive queries to separate the sessions after applying the time-threshold idea.

2. TWO SEARCH SYSTEMS

[3] `http://www.linkedin.com`



(a) Unsuccessful search.



(b) Successful search.

Figure 2: Traditional search on the UCI people directory (http://directory.uci.edu).

In this section, we explain the functionality and the user interface of an instant-search system and a traditional search system on the same data. We will use them as a testbed to develop solutions throughout the paper.

2.1 UCI Directory Search

The UCI Directory Search (http://directory.uci.edu) is a traditional search system on the university's people directory, which relies on MySQL full-text search functionality. Figure 2 shows its search interface, which sends a request to the server when a user clicks the "Search" button or presses the "Enter" key. The system is capable of supporting prefix (i.e., wildcard) queries, but cannot handle any typographical errors in a query. This search system returns results only that match the query keywords exactly. As shown in the figure, the query string "`debra lyon`" does not return any results due to a typo in the query.

2.2 PSearch

PSearch is a search system we developed and deployed more than four years ago. It supports instant, error-tolerant search on the same directory. A screenshot of its interface is shown in Figure 1, in which a user typed in the query string "`professor wenkata`". Even though the user did not complete typing in the second keyword, the system still found relevant search results. Notice that the two query keywords can appear in different fields of the records. The system treats each query keyword as a prefix, and highlights the matched prefixes. The system also does *fuzzy matching*, i.e., it can find records with keywords that are similar to the query keywords, such as a person name "`venkatasubramanian`". The similarity between strings is based on edit distance. The feature of supporting fuzzy search is especially important when the user has limited knowledge about the people they are looking for. As the user types in more characters, the system interactively searches on the data, and updates the list of relevant search results.

3. USER BEHAVIOR ANALYSIS

In this section, we propose a method for analyzing the query log of PSearch, in which every keystroke from a user triggers a new request to the server. Therefore, its log captures more information than traditional search logs, and show how users typed their queries. This unique feature allows us to analyze the user behavior and see if and at what point the user reformulated their query. This analysis is essential for a better understanding of the benefits of instant search.

3.1 Log Structure

We first explain the log structure and its key components. Due to the instant search nature of PSearch, an HTTP request is sent to the Web server for each keystroke, or a click on a result link. The Web server stores these HTTP requests with information such as IP, request time, query string, and the client browser type. Figure 3 shows some example log records. We now define the following related concepts.

Log Record: A *log record* is a line in the log containing a query string or a record link (i.e., URL) clicked by the user. For instance, each line in Figure 3 is a log record. A *click record* is a special type of log record containing a record link clicked by a user such as the last line in the figure.

Query: Every keystroke triggers a new request, which is logged as a query string. However, these logged strings are intermediate queries in the process of typing the complete query. We call these logged query strings simply as *queries*.

Session: In the literature, the concept of *session* has different definitions[10, 9]. In this paper, we define *session* as a sequence of queries by a user issued without a major interruption to fulfill a single information need, e.g., searching for a particular entity. The main reason to partition a query log into sessions is to separate each information need of a user. Whenever a user has an information need, they tend to be interactive with the system during a short period of time without any major pause until they are satisfied with the results or give up in frustration. Therefore, the time difference between two consecutive queries in a session needs to be within a certain time threshold such as 5 minutes. On the other hand, the user can have multiple information needs within the time threshold, and we cannot identify them as different sessions by time partitioning only. For this purpose, we use the edit-distance similarity of consecutive queries to identify different information needs. Most consecutive queries of the same user have a one-character difference due to the incremental typing behavior of the user, and they can be put into the same session. However, when the input search box is cleared and a new query has been started from scratch, or the length difference of two consecutive queries is larger than one, we end the current session and create a new session from that point. Let S_1 be the ended session and S_2 be the new session. We denote p_1 and p_2 as the longest

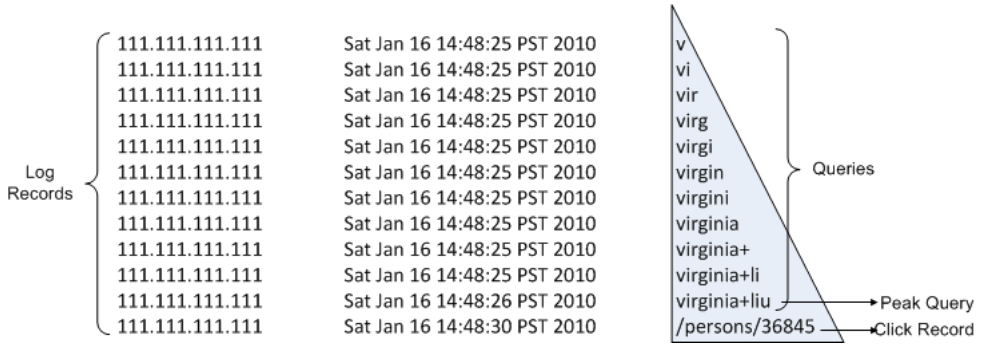


Figure 3: A small portion of PSearch log (with an anonymized IP address.)

query issued in session S_1 and S_2, respectively. When we decide to end S_2, we compare S_1 and S_2 to see if they have similar query keywords. We make this decision by looking at the normalized edit distance of p_1 and p_2, defined as:

$$ned(p_1, p_2) = \frac{ed(p_1, p_2)}{max(len(p_1), len(p_2))},$$

where $len(p_i)$ is the length of p_i $(i = 1, 2)$, and $ed(p_1, p_2)$ is the Levenshtein distance [8] between p_1 and p_2. We combine the sessions S_1 and S_2 if $ned(p_1, p_2)$ is smaller than a threshold (e.g., 0.5). Otherwise, we treat them as separate sessions with different information needs.

Peak Query in a Session: Queries in a session indicate how the user typed in keywords. In a sequence of keystrokes, some of them are more significant than others, because the results of these queries can change the user behavior. If a query returns the desired records, there is no need for the user to type more characters. On the other hand, if a query returns records different from the desired ones, the user will modify the query. We call such significant queries that have an affect on user behavior as *peak queries*. A peak query in a session should satisfy one of the following conditions:

1. It is the first query of the session and has more characters than the next query. For example, the query "`anastacia`" in Figure 4(c) is the peak query of the session, as it was likely copied and pasted by the user.

2. It is the last query of the session and has more characters than the previous query. For instance, "`virginia liu`" in Figure 4(a), and "`phil orw`" in Figure 4(d) are peak queries of their session.

3. It has more characters than its previous query and the next query. For example, "`phil techn`" is a peak query in the session of Figure 4(d).

Using the PSearch query log, we extracted over $350,000$ records issued by $2,800$ users within a one-year period, from September 2010 to September 2011. We partitioned these log records into sessions in three steps. First, we partitioned them using IP addresses to separate the queries of different users. Second, we grouped these partitions by looking at the time difference between consecutive queries. If the time difference was greater than 5 minutes, we put these queries into different sessions based on our session definition. Finally, we divided the partitions from the second step based on an edit-distance similarity of the queries in order to put all the similar consecutive queries into one session and separate the queries issued to search for different people.

In the next section, we analyze the sessions, and categorize them based on the user typing behaviors.

3.2 Session Patterns

Generally, the difference between two consecutive queries is usually only one character, since users usually type queries incrementally. This feature makes sessions have easily recognizable visual shapes. For instance, the session in Figure 3 has a triangular shape.

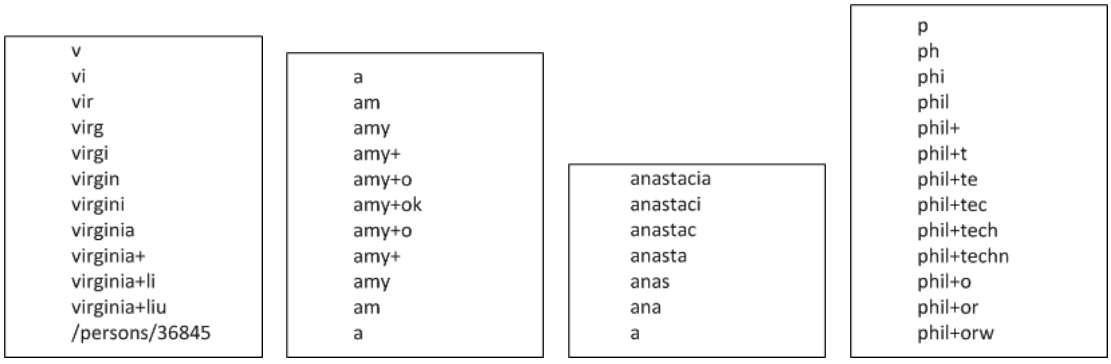
Analysis of the PSearch query log shows that there are several distinct categories of session patterns. The geometric shapes we observe are recurring and each shape has a specific explanation. Each pattern depicts a unique user typing behavior. Based on the geometric shapes of sessions, we categorize the sessions into four major categories. We now explain each of them in detail. (We name each pattern using a letter with a similar shape.)

L-Pattern (61%**)**: This pattern is seen when a user types in a query incrementally without a reformulation until finding the desired records in the results. In this category, either there is no spelling error in the query, or even if there is an error, the desired records still appear in the results because of the fuzzy matching feature of PSearch. Figure 4(a) is an example of this pattern. In this scenario, the user pressed the keys "`v`", "`i`", "`r`", "`g`", "`i`", "`n`", "`i`", "`a`", "", "`l`", "`i`", and "`u`" respectively and then clicked on one of the results. This pattern is the simplest, yet the most common user typing behavior. In our query log, 61% of sessions have this shape.

D-Pattern (7.2%**):** This pattern is very similar to the L-Pattern with one difference. The user types in a query as in the L-Pattern and then presses the backspace to clear the input search box to get ready for the next query. Figure 4(b) is an example of this typing behavior. The user pressed the backspace ten times after typing in the query "`amy ok`". 7.2% of sessions fall into this category.

Γ-Pattern (12.2%**):** This pattern occurs when the user starts by pasting a query copied from somewhere else and then optionally clears the input box for the next query. Figure 4(c) is an example of this typing behavior, in which the user pasted the word "`anastacia`" into the search box. This pattern covers 12.2% of the sessions in our query log.

B-Pattern (19.4%**):** This pattern shows that the user reformulates the query by deleting some characters and adding new keystrokes after a peak query. A peak point on each denticle is a peak query, which represents a point where the user either decides to reformulate the query or finds the tar-



(a) L-Pattern. (b) D-Pattern. (c) Γ-Pattern. (d) B-Pattern.

Figure 4: Session patterns.

geted records. Figure 4(d) is an example where, after typing "`phil techn`", the user decided that the keyword "`techn`" was not a good choice, and should be replaced by "`orw`". Thus, the user removed "`techn`" and typed "`orw`" instead. 19.4% of the sessions in the query log have this pattern.

The four patterns cover most of the sessions. The L-Pattern is the most frequent pattern. This fact shows that in most cases, the system can find the relevant results without requiring users to reformulate queries.

3.3 Log Statistics

We now report the statistics of the PSearch query log. As shown in Table 1, 96% of the log records are queries, while the remaining 4% are click records. There are far more queries than clicks, since each query represents a keystroke.

Title	Number
Number of distinct IP addresses	2,823
Number of log records	367,853
Number of queries	353,066
Number of clicks	14,787
Number of sessions	38,531
Number of sessions having at least one click	13,534
Median of session duration (seconds)	3
Average peak query length (characters)	8.89
Average number of queries in a session	9.16

Table 1: Statistics of the PSearch query log.

Using the partitioning scheme explained above, we obtained 38,531 sessions, 35% of which had at least one click. Most sessions had no clicks. However, this is not a sign of failure, because as seen in Figure 1, most of the information about a person such as phone number and email address can be found in the result page, which may fulfill the user's information need without an extra click.

The median of session durations is 3 seconds. The average peak query length is 8.89 characters, and there are 9.16 queries (i.e., keystrokes) on average in each session. If the peak query length is the same as the number of queries in a session, then this session has an L-Pattern and did not have a reformulation. If the number of queries is greater than the average peak query length, then there were reformulations in these queries, and on average a user typed more characters than the actual length of the final query. The smaller the difference between these two measures, the more successful the search system is, because the system returned the desired results with fewer reformulations by the user.

4. BENEFITS OF INSTANT SEARCH

In this section, we study the benefits of instant search. We compare PSearch with the traditional UCI search system. We focus on success rate, user typing effort in a session, and time spent per session. We extracted log records from the UCI directory search log and partitioned them in the same way we partitioned the PSearch log. We obtained more than 300,000 sessions from the log issued between December 2010 and March 2011. We used these records to do the analysis, and summarized the comparison results in Table 2.

	UCI D. S.	PSearch
Success Rate	71%	92%
Avg number of characters in a session	8.8	5.9
Median of session duration (seconds)	5	3

Table 2: Comparison of instant search and traditional search.

4.1 Success Rate

First, we compare how successful these two systems are in returning the intended results. To understand the satisfaction of the user in each system, we conducted a user study with a special interface asking users to give feedback for each session. We asked 18 users to search for people they know at UCI, and give feedback by clicking the "`Yes`" link (found) or the "`No`" link (not found) on the box in Figure 1 when they finish searching for a person. At the end of this study, for PSearch we collected 74 sessions with the user feedback, where 68 of them were "`Yes`" resulting in a 92% success rate. Similarly, for UCI Directory Search we collected 70 sessions from the same 18 users. In 50 of the 70 sessions, the user indicated "`Yes`", resulting in a 71% success rate. The user study shows that this instant search system is more successful than the traditional search system on fulfilling information needs of users. The main reason is due

to the fuzzy-search feature of PSearch, which can tolerate small typographical errors in a query and find the results in spite of these errors. Moreover, the instant-search nature of PSearch also guides users during the typing process. Seeing the similar keywords on the returned results helps users better formulate their queries.

Traditional search systems return matching records only if the user correctly types the complete query keywords. If the user is not sure about their information need, e.g., if they do not know the correct spelling of the name they are searching for, they will probably make mistakes during typing. If after a few tries the system still does not return relevant results for a query, the user may give up in frustration.

4.2 User Typing Effort in a Session

One way to compare the two systems is to measure how much effort is needed by the user to type in a query. We can estimate the effort by calculating the average number of characters in a session. Typing a few extra characters on a keyboard may not require a lot of effort for a user. Users might complete their keywords even though the desired results are already found by the system, especially when they look at the keyboard instead of the results during the typing process. Typing fewer characters to find the desired records becomes more crucial in mobile devices because of the *fat-finger syndrome*. Typing in mobile phones is an error-prone process, and it can be very annoying for users. Therefore, being able to return relevant results with the least effort is important for a search system for mobile devices. We set up an experiment to measure how quickly PSearch system returned relevant results. For this experiment, we extracted the sessions containing a query and a click from the UCI directory log. In these sessions, a click record in a session indicates the intention of the users for the query typed in the same session. To see how quickly PSearch found the same intended record for each session, we issued the same query character by character, simulating the typing process, until the intended record appeared among the first 5 results. This experiment showed that PSearch can find the intended records in 5.9 characters on average, 2.9 characters less than the 8.8 characters needed in the UCI directory search.

4.3 Time Spent in a Session

Another metric is the average amount of time spent to answer an information need. In the PSearch log, the queries were stored with their timestamps starting from the first keystroke in the session. Therefore, we can know when a user started and ended a session, and compute the time spent in a session. The only sessions whose durations cannot be computed are the ones where users pasted their queries, found their desired records, and left the system, because there is only one log record stored for each of these sessions. On the other hand, in the UCI directory search log, the first log record in a session is the first submitted query. Hence, we do not know how much time the user spent typing in the first submitted query. If there are multiple queries in a session, the time difference between the first query and the last query shows the time spent on reformulating the query and browsing the results. However, to be able to compare the session duration of the two systems, we need to estimate the time spent for typing the first query in the traditional search. We can estimate this time using the length of the first query, and an average typing speed. The average typing speed to enter text and make corrections was found as 32.5 words per minute, which corresponds to 2.7 characters per second by Karat et al [6]. Based on this average number, we computed the duration of each session in the UCI directory search log and found the median of these durations as 5 seconds. It shows that PSearch can save about 2 seconds per session compared to the traditional search.

5. CONCLUSION

In this study we studied the problem of how to analyze query logs of instant-search systems. We analyzed the query log of an instant-search people-search system at UCI and identified the user behaviors on such a system. We compared this system with a traditional search system on the same data. The comparison showed that instant search shortens the search time, and helps users find answers even if they only have partial knowledge of what they are looking for. The comparison also shows that instant search can save typing effort by returning relevant answers before the user completes typing the query. This feature is especially helpful for users accessing information from mobile devices, where each screen tap is time consuming and error prone.

6. REFERENCES

[1] http://www.google.com/insidesearch/instant-about.html.

[2] R. Baeza-Yates, G. Dupret, and J. Velasco. A study of mobile search queries in japan. In *WWW*, 2007.

[3] S. Bhatia, D. Majumdar, and P. Mitra. Query suggestions in the absence of query logs. In *SIGIR*, pages 795–804, 2011.

[4] B. J. Jansen, A. Spink, J. Bateman, and T. Saracevic. Real life information retrieval: a study of user queries on the web. *SIGIR Forum*, 32:5–17, April 1998.

[5] M. Kamvar, M. Kellar, R. Patel, and Y. Xu. Computers and iphones and mobile phones, oh my!: a logs-based comparison of search users on different devices. In *WWW*, pages 801–810, 2009.

[6] C.-M. Karat, C. Halverson, D. Horn, and J. Karat. Patterns of entry and correction in large vocabulary continuous speech recognition systems. In *SIGCHI*, pages 568–575, 1999.

[7] Y. Kim, J. Seo, and W. B. Croft. Automatic boolean query suggestion for professional search. In *SIGIR*, pages 825–834, 2011.

[8] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.

[9] H. C. Ozmutlu and F. Çavdur. Application of automatic topic identification on excite web search engine data logs. *Inf. Process. Manage.*, 41:1243–1262, September 2005.

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

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

[12] W. Weerkamp, R. Berendsen, B. Kovachev, E. Meij, K. Balog, and M. de Rijke. People searching for people: analysis of a people search engine log. In *SIGIR*, pages 45–54, 2011.