Characterizing the Usability of Interactive Applications Through Query Log Analysis

Adam Fourney
afourney@cs.uwaterloo.ca

Richard Mann
mannr@uwaterloo.ca

Michael Terry
mterry@cs.uwaterloo.ca

David R. Cheriton School of Computer Science
University of Waterloo

ABSTRACT

People routinely rely on Internet search engines to support their use of interactive systems: they issue queries to learn how to accomplish tasks, troubleshoot problems, and otherwise educate themselves on products. Given this common behavior, we argue that search query logs can usefully augment traditional usability methods by revealing the primary tasks and needs of a product's user population. We term this use of search query logs *CUTS*—characterizing usability through search. In this paper, we introduce CUTS and describe an automated process for harvesting, ordering, labeling, filtering, and grouping search queries related to a given product. Importantly, this data set can be assembled in minutes, is timely, has a high degree of ecological validity, and is arguably less prone to self-selection bias than data gathered via traditional usability methods. We demonstrate the utility of this approach by applying it to a number of popular software and hardware systems.

Author Keywords

Query log analysis, Usability

ACM Classification Keywords

H.5.2 Information Interfaces and Presentation: Miscellaneous

General Terms

Human Factors

INTRODUCTION

People rely on search engines (e.g., Google[1], Yahoo![2], Bing[3], etc.) to support their use of interactive systems [5, 6]. For example, users submit search queries to locate tutorials, troubleshoot problems, or learn how to use specific features of an application. Given this behavior, search engine query logs serve as centralized repositories cataloguing the day-to-day needs of the user base of any publicly available interactive system.

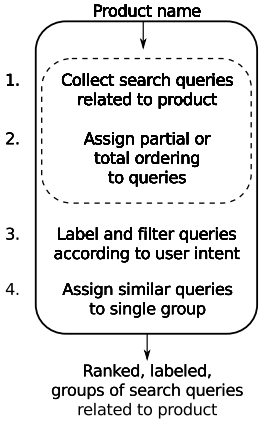

Figure 1. An overview of CUTS. Steps 1-2 are easily performed with access to raw query logs, but otherwise require approximation techniques. Step 3 utilizes our query taxonomy specialized for interactive systems.

In this paper, we argue that search engine query logs can be filtered and transformed into forms that usefully complement and augment data collected via traditional usability methods. We demonstrate this potential by introducing an automated process for harvesting, ordering, labeling, filtering, and grouping search queries to understand the common tasks and needs of a user base (Figure 1). We call this process *CUTS*—characterizing usability through search. Importantly, the labeled, ordered data produced by CUTS can be assembled in minutes, is timely, has a high degree of ecological validity, and is arguably much less prone to self-selection bias than traditional means of collecting data from users.

As an example of the utility of this approach, an approximation of this process can be illustrated using Google Suggest, the service that provides query completion suggestions for a given input. Given the phrase "firefox how to", Google Suggest produces a list of 10 suggested completions (Figure 2). As we will show later, these suggestions closely correspond to the 10 most popular queries matching that input.

From the list of top 10 Firefox "how to" suggestions (Figure 2), it is immediately clear that users have a number of pri-

[1] http://www.google.com
[2] http://www.yahoo.com
[3] http://www.bing.com

CHI 2011, May 7–12, 2011, Vancouver, BC, Canada.

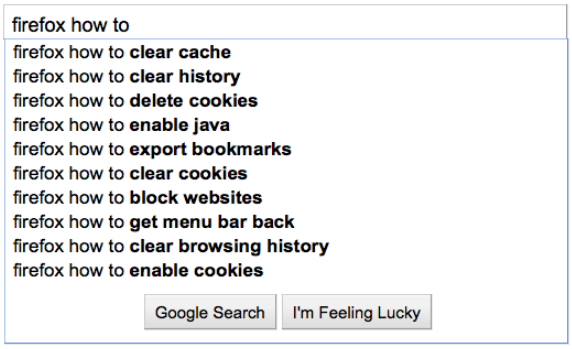

Figure 2. The top 10 suggestions provided by Google Suggest for the phrase "firefox how to".

vacy and security concerns, as evidenced by their desire to clear their cache, history, and cookies. However, the eighth item ("get menu bar back") is particularly interesting. An inspection of the Firefox user interface (version 3.6 on Windows), reveals that the top-level menu bar is easily hidden by deactivating the "Menu bar" item in Firefox's "View → Toolbars" sub-menu. However, once this action is taken, it is not easily reversed: The top-level menuing system is now hidden, removing the very means the user would employ to attempt to re-instate the menu bar. What is noteworthy about this example is that we quickly moved from data derived from query logs to a testable hypothesis regarding the usability of the software.

The contributions in this paper lie in expanding this manual process to the automated one shown in Figure 1. While seemingly straightforward, automating this process requires overcoming a number of challenges: Raw query logs are not made publicly available; there is a need to automatically determine query intent for the purposes of labeling and filtering queries (for example, to distinguish troubleshooting queries from those seeking to download the application); and differently phrased queries on the same topic should be reduced to a common canonical form. Our specific contributions, outlined below, address these challenges.

To address the problems of obtaining and ranking search queries, we demonstrate how publicly available query suggestion services (e.g., Google Suggest) and web-based tools for advertisers can be employed to create reasonable approximations of raw query logs.

We also introduce two new query classification schemes to address the need to label and filter queries. The first classification scheme is a taxonomy that extends previous search query taxonomies to include categories relevant to interactive systems. For example, this new taxonomy differentiates between queries issued to troubleshoot a problem and those seeking a tutorial. The second classification scheme considers *how* a query is phrased. We show that how a query is phrased closely corresponds to the categories of our specialized taxonomy. CUTS exploits the relationship between these two classification schemes to ascribe query intent from query phrasing.

Finally, common questions or issues are often expressed using a number of different query phrasings. To cope with this variability, we introduce a transformation that enables minor differences between queries to be ignored.

The rest of this paper is structured as follows. We first present related work, then describe our method for harvesting and ranking search queries using publicly available services. We then introduce our two classification schemes and show how they can be used to label and filter search queries. The final step of the process, grouping queries, is discussed, and a set of strategies are introduced to assist with this process. We then present a series of examples illustrating the overall utility of this approach, and conclude with a discussion of the limitations of the technique.

BACKGROUND & RELATED WORK

In recent years, researchers have demonstrated the potential for search engine query logs to model and predict real-world phenomena and events. For example, Jeremy Ginsberg *et al.* have demonstrated how query logs can be employed to help track the spread of influenza over time [10]. In this latter research, "health-seeking behaviour" is automatically detected by monitoring search terms associated with influenza (symptoms, medications, etc.). This allows the Google Flu Trends application[4] to estimate the prevalence of influenza infections on a week-to-week basis. The resultant models closely agree with data released by the Center for Disease Control (CDC), though they exhibit much less lag: Models built using query logs show a 24 hour lag in tracking flu trends, compared to the week lag of the CDC.

More generally, Richardson [24] argues that query log analysis could quickly become an indispensable tool for researchers working in such human-centric fields as anthropology, sociology, psychology, medicine, economics, and political science. He notes that query logs function as if "a survey were sent to millions of people, asking them to, every day, write down what they were interested in, thinking about, planning, and doing." Accordingly, he argues that "taken as a whole, across millions of users, ... queries constitute a measurement of the world and humanity through time" [24]. To demonstrate his point, Richardson describes a common search pattern that unfolds over the course of three to six months, starting with a user's search for "mortgage calculators". Within a week, these same users search for "realtors". About one month later, they search for legal services (e.g., "notary"), and three months later, their searches include those for home furnishing (e.g., "pottery barn"). As with Google Flu Trends, this latter example shows the potential for query logs to describe real-world phenomena.

Within the realm of interactive systems, the research literature contains many accounts of search query logs being used to improve *information interfaces*—interfaces in which finding or accessing information is a user's primary task. For example, Zhicheng Dou *et al.* demonstrate how query logs can be used to improve personalized search [8]. It is also common for website designers to use query logs to help de-

[4] http://www.google.org/flutrends/

termine what links should be provided in their site's top-level navigation [23]. Our work broadens these previous uses of query logs, demonstrating their utility in understanding users' needs with any publicly available interactive system.

While prior work in interactive systems has focused on using query logs to improve information interfaces, work by Brandt *et al.* has demonstrated how software developers employ search engines when creating software [6]. Through studies of developer practices, Brandt *et al.* show that software developers make extensive use of Google for a variety of programming-related tasks, such as using search to translate knowledge from one domain to another (e.g., from one programming language to another). Our work more generally considers how people employ search to support their use of interactive systems.

Importantly, all of the aforementioned research has been conducted by institutions or companies with direct access to search query logs. Access to these query logs is highly guarded, especially after the privacy problems encountered when AOL released (what they felt was) an anonymized sample of their query logs [28]. Lacking direct access to raw query logs, Bar-Yossef and Gurevich have demonstrated how statistics of these logs can be approximated using an importance sampling technique [4]. This technique estimates the popularity of certain keywords by using parameterized models (derived from the aforementioned AOL search query logs) and by sampling query completions provided by query completion suggestion services. Our work is inspired by this research, as we also use query completion suggestion services. However, we perform more exhaustive searches of the query suggestions for a specific topic (i.e., a specific interactive system), and we supplement query completion suggestions with data harvested from web-based advertising tools.

There are also a number of research endeavors which are related to CUTS, but which do not use query logs as a primary data source. In particular, many researchers have explored how logs of user interface events can be leveraged to learn about a system's usability (surveyed in [14]), or to help characterize a system's user community (e.g., [18]). In this space, David Akers *et al.* [1] have recently demonstrated that invocations of the *undo* and *erase* commands in 3D modelling software can serve as indicators of usability problems. Similarly, Amy Hurst *et al.* have shown how low-level mouse data, related to menu use, can help determine if a user is having difficulty locating an item of interest [16]. We view query log analysis as complementary to the analysis of user interface event logs, and as providing a higher-level view of the issues encountered by users. Moreover, query logs tend to more directly showcase issues regarding feature discoverability, as evidenced by the "firefox how to get the menu bar back" example noted in this paper's introduction.

Finally, online social media sites such as blogs, customer support forums, and sites hosting customer product reviews are other possible sources for understanding user needs. These sources are heavily utilized in sentiment analysis research (also known as "opinion mining") [22]. Sentiment analysis seeks to determine, from a person's writing, his or her attitude towards a particular topic. When applied to the evaluation of interactive systems, sentiment analysis can determine if a product's reviews are generally positive or negative, and can determine which aspects or features of a product typically lead to a positive or negative review. While sentiment analysis is a promising area of research, its effectiveness and scope as a tool for understanding user needs may be impacted by the effort required for users to post content to these social media outlets (compared to simply performing a web search). Conversely, query log analysis provides a view of the issues encountered by the broader user community who may not be regular contributors to the online discussion of interactive systems.

In summary, previous work demonstrates the general utility of query log analysis: it yields timely, highly ecologically valid data that can quickly lead to significant insights when studying a wide range of phenomena. However, query log analysis has not been previously applied to the problem of characterizing the overall usability of interactive systems. In the rest of this paper, we demonstrate its specific utility in the realm of interactive systems by detailing each step of the CUTS process.

QUERY HARVESTING

The core of the CUTS process lies in the analysis of search query logs. When access to raw query logs is not possible, search queries can be harvested using publicly accessible interfaces: Modern search engines provide indirect and privacy-preserving access to their logs through their query completion suggestion services [4]. In this section, we describe a process for systematically harvesting queries related to a particular interactive system using these services. We also provide evidence that the results of this method can be considered a representative sampling of the raw query logs.

Harvesting from auto-completion services

Query completion suggestion services operate as if backed by a prefix tree [4]. When viewed in this way, the characters making up a partially entered query define a path through the tree starting at the root, passing through numerous nodes. Each node contains a listing of popular queries whose prefix matches the path taken thus far. Query completion services follow the paths prescribed by partially entered queries, and return the suggestions listed at the ends of these paths.

Given the tree-like structure of these services, a standard depth-first or breadth-first tree traversal can be performed by expanding partial queries one character at a time, starting with the name of the system under investigation (Figure 3). A leaf (or external node) is reached when the completion service returns no suggestions for the given prefix.

```
firefox,
firefox a, firefox aa, firefox aaa, …
…
firefox z, firefox za, firefox zaa, …
```

Figure 3. Input sequence representing a depth-first traversal of Google's prefix tree rooted at "firefox".

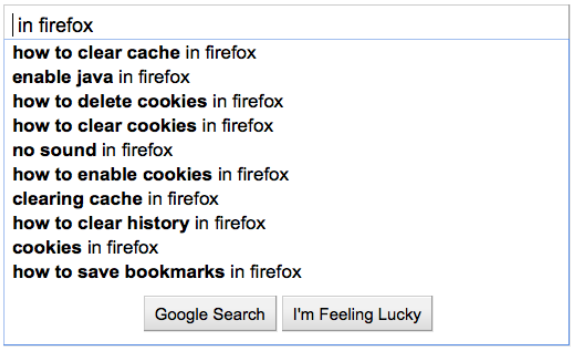

Figure 4. The *Google Suggest* auto-completion service varies its suggestions based on the caret position.

Mining additional queries

Some search providers, such as Google, vary their query suggestions depending on the position of the caret in the search query input box (Figure 4). More specifically, Google provides a list of the top 10 completions that either begin or end with the phrases on the left or right side of the cursor. Given this behaviour, the whole tree traversal procedure can be repeated to uncover query suggestions that end with a particular suffix, providing a more complete sampling of the query logs.

By executing a systematic search of the query completion tree, many queries can be collected for a given topic. For example, on June 19th, 2010, we recorded 74,795 unique queries when performing a systematic search for queries incorporating the term "Firefox" in Google Suggest's query auto-completion database. Similar results were obtained for other systems for which we collected data (Table 1).

Application	Description	# of Query Suggestions
iPhone	A smartphone platform	476,462
Ubuntu	A Linux distribution	122,242
Photoshop	An image editor	119,791
Firefox	A web browser	74,795
Chrome	A web browser	45,499
Kindle	An eBook reader	21,621
Gimp	An image editor	14,569
Inkscape	A vector graphics editor	2,501

Table 1. Number of unique query suggestions provided by Google for various interactive systems.

Representativeness and timeliness of auto-completions

In harvesting these queries, our working assumptions are that (1) query completion services are derived from the raw query logs, (2) a given query's prevalence in these logs will have some bearing on its ranking in the list of suggestions, and (3), the suggested completions are *timely*. By "timely," we mean that query completion services assign more weight to queries performed within a recent window of time. In the following subsections, we briefly provide evidence that these assumptions are sufficiently valid for our purposes.

Representativeness of query completion suggestions

In the case of Google, some information about their query suggestion service has been published [11]. Specifically, Google's documentation notes that "All of the queries shown in (Google) Suggest have been typed previously by other Google users". Google also states:

> *Our algorithms use a wide range of information to predict the queries users are most likely to want to see. For example, Google Suggest uses data about the overall popularity of various searches to help rank the refinements it offers.* [13]

Later, when discussing the use of marketing tools to rank queries, we will provide further evidence that auto-complete suggestions have a close correspondence to the most frequently, recently performed queries.

Timeliness of query suggestions

To identify trends and new issues as they arise, it is desirable that query suggestion services emphasize recent searches over those performed in the more distant past. To study the timeliness of Google's query suggestion service, we monitored the query completion suggestions for a range of products and software applications for a period of approximately three months (June 2010 through August 2010, inclusive). Suggestions were sampled on Monday, Wednesday, and Friday on each week during this timeframe. An analysis of the collected data reveals that Google updates its auto-completion database approximately once every 14 days. These results indicate that Google is actively maintaining its query suggestion database.

Knowing the frequency with which these services are updated is advantageous, but is not sufficient for determining the extent to which current search trends are represented in query suggestions. To investigate this question, we can examine when a noteworthy event begins to appear in query suggestions. A prime candidate for exploring this question is provided by the release of the iPhone 4 on June 24th, 2010. Almost immediately, there were reports of significant signal degradation when the phone was held in a certain way [9]. The first evidence of this issue was spotted in the query suggestions on July 14th, 2010. On this date, the partial query "iphone d" resulted in Google suggesting "iphone death grip", while "iphone a" yielded "iphone antenna", and "iphone how to h" yielded "iphone how to hold". None of these queries appear in the suggestions sampled on previous dates. This corresponds to a lag of about 20 days, suggesting that the query completion services place sufficient weight on recent queries.

RANKING QUERIES

After harvesting queries, the next step is to assign an importance rank to each query. When queries are sampled from query suggestion services, detailed query frequency information is not made available (current services do not indicate how often each search is performed). We substitute this missing data in two ways. First, we complement our data set with data collected from advertising and market research tools, such as the Google AdWords Keyword Tool [12]. Second, we examine the structure of the synthesized prefix tree to obtain a *partial ordering* of the queries not covered by the market research tools. We describe each technique in turn.

Using marketing tools to assign ranks

Google provides a set of tools that can be directly applied to the problem of ranking queries. The Google AdWords Keyword Tool [12], intended to help marketers valuate keywords for advertising purposes, can be configured to report the estimated average global monthly search volume for any exact phrase. As such, it is possible to directly rank many query suggestions using this tool alone. In doing so, we again find that there is good correspondence between the harvested queries and their estimated search volume.

While many queries can be directly ranked using the Google AdWords tool, not all queries can be ranked in this way; Google AdWords provides no data for queries whose monthly search volume is below some threshold, and this threshold is reached well before the list of query suggestions has been exhausted. For example, on June 19th, 2010, we harvested 74,795 unique query suggestions for the Firefox web browser. However, Google AdWords provides search volume data for only 15,057 of those queries. In short, the search volume of about 80% of the Firefox queries falls below the threshold reported by AdWords. Accordingly, we must employ another means of ranking the remaining queries, as described next.

Generating a partial ordering

While query suggestion services do not return the frequency with which each suggested query is performed, we have argued that they operate by returning the most popular queries for a given input. We can use this behaviour to derive a *partial ordering* of the query suggestions. The key insight is this: We know that *the 10 query suggestions returned for a given prefix are more popular than all other queries later harvested that also begin with that same prefix*. An example illustrates this point.

Returning to the earlier Firefox example, the suggestion "firefox menu bar missing" appears in Google's top 10 suggestions for the prefix "firefox m". Thus, we can infer that the "firefox menu bar missing" query is more popular than the 2362 other suggestions occurring in the data set that also share the prefix of "firefox m". We say that this query has 2362 *subordinates* in order to convey this relationship.

This ranking technique provides only a partial ordering because we can only perform comparisons of a node with its ancestors and descendants in the prefix tree. We *cannot* directly compare suggestions occupying separate branches of the tree. Nevertheless, for queries whose search volume falls below the AdWords reporting threshold, a search volume-based ranking will be crudely approximated by simply sorting those queries according to their number of subordinates.

These first two steps of harvesting and ranking queries provide us with a suitable, privacy-preserving, publicly accessible replacement for raw query logs. In the remainder of the paper, our technique assumes only that one has access to a ranked list of search queries relating to the interactive system of interest.

QUERY CLASSIFICATION AND FILTERING

Given a ranked list of queries, the next step is to automatically classify queries according to the likely intent of the individual performing the search (for example, to distinguish troubleshooting queries from those seeking to download the application). Once labeled, query logs can then be filtered to select entries that are potentially related to user tasks and usability issues.

Before queries can be automatically labeled and filtered, we first need to understand the range of system-related queries that users submit to search engines. While previous work has developed a number of taxonomies for general classification of search queries (e.g., to distinguish between navigational and information-seeking queries) [7, 17, 25], we found these taxonomies too broad for our purposes. Instead, a classification scheme specialized for the domain of interactive systems is needed. Additionally, we need to understand what features of a query can be used to support automatic labeling.

In this section, we address both of these needs: We introduce a taxonomy of *query intent* specialized for interactive systems, and a second classification scheme that describes how a query is *phrased*. As we will show, in this domain, query phrasing is strongly related to query intent. Based on the aforementioned intent-phrasing relationship, we present a set of heuristics for automating the process of labeling and selecting queries of interest.

Domain-specific query intent taxonomy

Following the basic methods of grounded theory [27], we developed our query taxonomy by performing open coding on 200 randomly sampled queries regarding the GIMP software application. From this initial coding, we identified a set of common, higher-level themes, which led to our taxonomy. The resultant taxonomy includes six separate classes of interactive system queries, synthesized from the perspective of query intent:

QUERY INTENT:

- **Operation Instruction**
  Would the query be used to find instructions for performing a specific operation?
- **Troubleshooting**
  Would the query be used for troubleshooting a bug or error?
- **Reference**
  Would the query be used to find reference material? (e.g., a list of keyboard shortcuts)
- **Download**
  Would the query be used to acquire, download, or install something?
- **General Information**
  Would the query be used to find general information about the application? (e.g., product reviews or comparisons)
- **Off-topic**
  Is the query unrelated to the software / project?

Query phrasing classification scheme

In parallel with developing the former taxonomy, we also developed a classification scheme that describes how individual queries are phrased. The motivation for developing this scheme arose during our open coding sessions: In considering the range of queries, it appeared that *how* a query was phrased was very much related to the *intent* of the user. As we will show, there is indeed a relationship.

Based on the open coding of the queries, the following high-level categories of query phrasing were identified:

QUERY PHRASING:
- **Noun phrase** (e.g., gimp brushes)
- **Imperative statement** (e.g., gimp rotate text)
- **Question** (e.g., how to draw a line in gimp)
- **Statement of fact** (e.g., gimp won't start)
- **Present participle** (e.g., rotating text in gimp)
- **Other**

In the next section, we show that raters are able to achieve a high degree of inter-rater agreement when using the intent and phrasing taxonomies to label search queries. This agreement lends support to the overall utility of the taxonomies as instruments for labeling search queries.

Inter-rater reliability of the classification schemes

To establish the inter-rater reliability of these two classification schemes, two researchers applied both schemes to a set of 195 queries sampled from the GIMP and Firefox datasets. The GIMP and Firefox datasets were collected from Google Suggest on May 23rd, 2010 and June 19th, 2010 respectively. Selection of the 195 sample queries proceeded as follows: For each application, the top 50 queries (by search volume) were selected, followed by an additional 50 randomly selected queries. The resulting set of 200 samples shared 5 queries in common with the set used for the initial open coding and were thus excluded from our validation process.

In labeling this data set, we achieved an overall inter-rater reliability rate of $\kappa_{\text{intent}} = 0.76$ for query intent, and $\kappa_{\text{phrasing}} = 0.79$ for query phrasing, using the Cohen's kappa measure of rater agreement. Inter-rater reliability across the 4 sources of queries is listed in Table 2. The observed agreements are considered to be substantial [19].

Before describing how the query phrasing classification scheme can be used to identify query intent, we first show how queries are distributed across these two classification schemes. These query distributions lend additional arguments for the overall utility of this approach.

Query Source	κ_{intent}	κ_{phrasing}
Firefox, top 50	0.74 (substantial)	0.80 (substantial)
Firefox, random 50	0.86 (near perfect)	0.80 (substantial)
GIMP, top 47	0.66 (substantial)	0.72 (substantial)
GIMP, random 48	0.66 (substantial)	0.81 (near perfect)

Table 2. Inter-rater reliability for each of the four sample sets.

	Rater 1		Rater 2	
Query Intent	Freq.	%	Freq.	%
Opr. Instr.	84	43%	80	41%
Troubleshooting	15	8%	17	9%
Reference	21	11%	19	10%
Download	55	28%	62	32%
General	12	6%	12	6%
Off topic	8	4%	5	2%

Table 3. Frequencies of query intent labels for the 195 randomly selected GIMP and Firefox queries harvested from Google Suggest.

Characterizing query data

The classifications of the 195 labeled queries are summarized in Table 3. The categories we find interesting for usability analysis coincide with the first two listed in the table and the taxonomy: "Operating Instruction", and "Troubleshooting". In our sample, about half of all query suggestions fall within categories that are of interest to HCI researchers and practitioners, demonstrating the overall richness of query logs when studying interactive systems.

Relationship between query phrasing and intent

If we compare how a query is classified in each scheme, we find that how a query is phrased strongly correlates with query intent. These findings are summarized in Table 4.

Query Intent	Noun	Imperative	Question	Fact	P. Participle	Other
Opr. Inst.	0.30 (38)	0.90 (28)	0.87 (13)		1.00 (5)	
Troubleshooting	0.05 (6)			1.00 (9)		
Reference	0.14 (18)	0.03 (1)	0.13 (2)			
Download	0.42 (53)	0.07 (2)				
General:	0.09 (12)					
Off topic						1.00 (8)

Table 4. Probability of query intent given its phrasing type based on the labels assigned by rater 1. Raw frequencies are listed in parentheses. Similar values are achieved using the labels assigned by rater 2.

There are a few noteworthy observations to make in this table. As can be seen, in our sample set, if a query is phrased as an imperative statement, there is a 90% chance that the query is seeking operating instructions. A similar probability (87%) applies if the query is phrased as a question. Finally, if a query is phrased as a statement of fact, then it is almost certainly being used for troubleshooting. These relationships provide us with a set of strategies for automating the labeling of queries, which we describe next.

Heuristics for automating query labeling

In the previous section, the relationship between query phrasing and intent was established by examining labels that were *manually* assigned to queries by a pair of human raters. Automating the CUTS process requires mechanization of the query labeling step. Through further inspection of the data, we have found that certain keywords or patterns are highly indicative of each of the different phrasing types. For example, queries containing the phrase "how to" indicate questions. Once a query's phrasing has been established, we can then infer its intent using Table 4. Here we focus on queries phrased as questions, imperative statements, and statements

Labels		Pattern	Example Query
Operating Instructions	Question	**how to ___ in** *SystemName*	how to delete history in firefox
		SystemName **how to ___**	firefox how to clear cache
		can *SystemName* **___**	can firefox block websites
		does *SystemName* **___**	does firefox have private browsing
	Imperative	**use** *SystemName* **___**	use firefox for windows update
		make *SystemName* **___**	make firefox default browser
		SystemName **set ___**	firefox set default zoom
		create ___ in *SystemName*	create a new profile in firefox
		SystemName **create ___**	firefox create pdf
Troubleshooting	Fact	*SystemName* **is / isn't ___**	firefox is starting slow
		SystemName **can / can't ___**	firefox can't add bookmarks
		SystemName **will / won't ___**	firefox won't open pdf
		SystemName **does / doesn't ___**	firefox doesn't play sound
		SystemName **has / hasn't ___**	firefox has no address bar

Table 5. Filtering templates for labeling the phrasing and likely intent of queries.

of fact because these phrasing types most reliably indicate searches for instructions or troubleshooting information.

A partial list of phrasing patterns is presented in Table 5. These patterns were generated through a manual inspection of labeled data, and serve as basic heuristics for labeling different types of queries.

Many queries will not match any pattern, and will thus go unlabeled at this stage of processing. In the next section, we describe a technique for grouping related queries. When queries are grouped, labels for the individual queries are extended to the group, increasing the coverage of the labeling.

GROUPING SIMILAR QUERIES

The final step in CUTS is to reduce the variability with which queries are expressed in the data set. In query logs, common questions or issues are expressed using a number of different query phrasings. As an example, GIMP users may search "how to draw a circle in gimp", or they may simply type "gimp draw circle". Given this variability, it is desirable that similar queries be grouped, and their weights or rankings combined, in order to better estimate the prevalence of a given issue.

To group similar queries, we transform queries to a canonical form where inconsequential differences are ignored (e.g., see Table 6). This transformation applies the following rules:

- Convert inflected word forms to common word lemmas. We use the WordNet lexical database [20] to perform this transformation (e.g., "deleting cookies" becomes "delete cookie")
- Remove all instances of stop words (such as "and", "the", "to", "but", etc.)
- Remove words devoid of alphabetic letters (e.g., "3.6.10", and other non-English strings)
- Sort the query terms alphabetically.

Using this technique, it is possible to achieve a modest reduction in the size of the data set. As an example, the Firefox data set of 74,795 unique queries is represented by 39,435 canonical query groups (53% of the original size). More importantly, we found that a canonical group's cardinality (number of distinct query phrasings) is directly related to the popularity of the group's overall topic or concern; compared to less popular topics, those experiencing high search volume yield logs that contain a more complete sampling of the alternative phrasings with which those queries can be expressed. Consequently, those high-volume queries tend to form groups of higher cardinality. To illustrate this point, Table 7 lists the cardinality of the canonical groups associated with the top 10 "firefox how to" queries already mentioned in the Introduction. All but the last of these queries fall within the top 99.6th percentile of group sizes, thus reinforcing the popularity of these concerns.

"firefox lost toolbar"	
lost my toolbar firefox	firefox toolbars lost
lost firefox toolbar	firefox lost my toolbar
lost all toolbars in firefox	firefox lost all toolbars
lost toolbar in firefox	firefox lost toolbar
lost my toolbar in firefox	lost my firefox toolbar
firefox toolbar lost	

Table 6. 11 distinct queries which share the canonical representation "firefox lost toolbar".

"Firefox how to" ...	Canonical Form	Cardinality of group
clear cache	cache clear	110
delete cookies	cookie delete	60
clear cookies	clear cookie	44
enable java	enable java	41
export bookmark	bookmark export	40
enable cookies	cookie enable	32
clear history	clear history	30
block websites	block website	29
get menu bar back	back bar get menu	16
clear browsing history	browse clear history	5

Table 7. Canonical groups associated with the top 10 "firefox how to" queries.

FINAL OUTPUT OF CUTS

The output of CUTS is a categorized and ranked list of query groups relating to the system under investigation. A sample of this output, for the Firefox application, is presented in Table 8. The final ranking of groups is determined by summing the search volumes of each group's member queries, and then sorting those groups accordingly. When search volume information is not available, a sum of subordinate counts is used instead.

Groups Containing Opr. Instr. Queries	Groups Containing Troubleshooting Queries
cache clear	not respond
clear cookie	not open pdf
cookie delete	slow
block website	crash
cookie enable	mode safe
proxy	check not spell
delete history	constant crash
...	...

Table 8. Query groups, related to Firefox, output after query harvesting, ranking, labeling, filtering and grouping.

APPLICATIONS

In this section, we apply our technique to a number of different interactive systems. Our goal here is to demonstrate the wide range of insights that can be gained using this approach. We structure this section by showing how issues related to language, desired functionality, and poor affordances can all be detected using this technique.

"Speak the user's language"

Query logs provide an excellent view of the vocabulary and terminology with which users conceive their use of interactive systems. However, this terminology does not always match that which is used by their systems. When such discrepancies arise, the associated systems can be considered to be in violation of Jakob Nielsen's "Speak the User's Language" usability heuristic [21]. We provide two examples of this problem that we identified using our technique.

Black and white, but not grayscale

The GNU Image Manipulation Program (GIMP) is an open source raster graphics editor offering similar functionality to Adobe's Photoshop application. On May 23rd, 2010, we harvested 14,559 queries relating to this software application. Analysis of the GIMP data set reveals that the terms "black" and "white" co-occur in 93 distinct queries. In each case, the queries inquire about converting color images to black and white. According to the Google AdWords tool, the query "gimp black and white" is searched an average of 590 times a month, or about once every 74 minutes.

Inspecting GIMP's interface (version 2.6) reveals that there are at least three alternative methods for converting a color image to grayscale. These methods are labeled as "grayscale", "desaturate", and "channel mixer". Such technical terms may not be familiar to a sizeable portion of GIMP's user base, as evidenced by the vocabulary used in the harvested queries. Given this finding, one could create a "black and white" command that aggregates into one command the many methods of transforming a color image into a grayscale image.

Clip, but not crop

Inkscape is an open source vector graphics editor similar to Adobe's Illustrator program. On May 22nd, 2010, we harvested 2,501 queries relating to Inkscape. Interestingly, the 8th highest volume query was "inkscape crop", with an average of 480 searches performed each month. However, being a vector graphics application, Inkscape does not have a "cropping" tool; cropping is specific to raster graphics. The equivalent operation for vector graphics is to "clip". This very popular query suggests that new Inkscape users are relying on Google to translate knowledge from one domain (i.e., raster graphics) to another domain (i.e., vector graphics). This behaviour closely resembles similar behaviour exhibited by programmers' use of Google [6]. Recognizing this issue, Inkscape could provide a "crop" command or a help entry that assists users in setting the clip for their document.

Desired functionality

In addition to identifying potential usability issues related to terminology, we found query log analysis to be an excellent source for discovering desired functionality.

Blocking unwanted calls

One popular class of queries related to Apple's iPhone product inquires about the possibility of selectively blocking unwanted calls from specific telephone numbers. While this feature is not currently supported by the device, users search for information on performing this task at least 5,800 times a month, or once every 7.5 minutes. The consensus among the user community is that the issue can be resolved by associating a silent audio clip as the ringtone of unwanted telephone numbers. That this issue is so popular suggests users would be well-served if provided with a sanctioned means of achieving this same behaviour.

Changing screen savers

Another example of identifying desired functionality emerges when analyzing the searches specific to Amazon's Kindle eBook reader. Specifically, query log analysis reveals 89 distinct phrasings of the query "how to change your kindle screensaver". The Kindle device ships with a few dozen stock images that are displayed by the device when not in use. However, these images cannot be customized by the end user. Again, the popularity of these searches suggests that such a feature would be welcomed.

Drawing shapes in GIMP

Finally, an analysis of the GIMP query data set reveals many queries related to drawing primitive shapes: Roughly 130 unique queries inquire about drawing various types of lines, 80 unique queries inquire about drawing circles, 40 queries inquire about drawing rectangles, 20 queries inquire about drawing squares, and 14 queries inquire about drawing ellipses. Moreover, the suggestions "gimp how to draw a line", and "gimp how to draw a circle" appear in the top 10 suggestions for the prefix "gimp how to", and the Google AdWords tool reports that the query "gimp draw circle" is performed an average of once an hour, each and every day. These queries are noteworthy because GIMP provides no explicit tools for drawing simple shapes. Dedicated tools for these functions would likely find great use by GIMP users.

The usability cost of poor affordances

As a final example of the types of problems that can be uncovered using query log analyses, we consider the usability cost of poor affordances and uninformative error messages.

Ubuntu's "authentication failure"

Ubuntu is currently one of the most popular GNU/Linux distributions. For reasons of security, Ubuntu disables the "root" superuser account by default, requiring users to issue the "sudo" command to gain superuser privileges. The root account has otherwise been present and used in UNIX and UNIX-like systems for decades.

While Ubuntu's policy is arguably a positive change for security, the operating system may not be adequately com-

municating this policy to new users: Attempts to log in as the root user (in Ubuntu version 10.04) simply result in an "authentication failure" error message. An analysis of the queries related to Ubuntu reveals nearly 130 distinct query phrasings all asking about how to access the root user account. The specific query "ubuntu login as root" is performed 720 times a month, or about once an hour. Similarly, a search for the error message "su authentication failure ubuntu" occurs about once every 7 hours. These findings suggest that users would be well served by a more helpful or detailed error message which could communicate the proper course of action when attempting to login as the root user.

DISCUSSION

In this section, we more broadly discuss issues related to using query logs to understand the needs of users of interactive systems. We begin by comparing the output of CUTS to manually curated "frequently asked questions" (i.e., FAQs). We then discuss how query log analysis can factor into existing usability practices, and enumerate various issues that may affect the rankings produced by our method.

Comparing CUTS' output to published FAQs

CUTS reveals search queries that are frequently performed by a system's users. As such, its output is directly comparable to lists of frequently asked questions (FAQs) commonly provided as documentation for many software applications. However, standard FAQs are curated by individuals, and require continual maintenance and individual judgement regarding inclusion of content. Accordingly, we expect CUTS to more accurately represent the needs of the user base over time. A comparison of our results with the GIMP FAQ lends support to this notion.

The GIMP FAQ[5] contains 25 questions/answers in the section entitled "Using GIMP." Sixteen of these issues overlap the top issues found using CUTS. The FAQ issues that don't overlap with our results tend to be quite specialized (e.g., "How do I configure X server to do global gamma correction?"). Since GIMP's user base primarily consists of casual users who perform relatively simple tasks [18], very few users will benefit from the answers to these specialized questions. In contrast, CUTS reveals a more representative set of questions related to the simple tasks users have been found to perform (e.g., "gimp how to convert to black and white").

We have also compared CUTS' results to the Firefox knowledge base (KB)[6]. Again, the CUTS data suggests many potential usability issues not directly addressed by the Firefox KB (e.g., "where do firefox downloads go"). The key point in both instances is that CUTS provides a data-driven view of user concerns that is continuously updated.

Integrating query log analysis in usability practices

Throughout the paper, we have been careful to note that query logs can be used to identify *potential* usability problems of interactive systems. While a query may *suggest* that users are experiencing difficulties with a particular aspect of the system (e.g., "gimp how to draw a circle"), further details and context are required before one can conclude the nature and severity of the issue. This additional information can be obtained using standard evaluation techniques involving users or expert evaluators. Since many methods (e.g., cognitive walkthrough) require representative tasks to be identified for evaluative purposes, CUTS can assist by supplying a ranked list of common tasks and needs.

A ranked list of common queries can also be used to assign importance to existing lists of known usability issues. The benefit of using the results of CUTS is that this ranking is derived from the search behaviour of thousands, if not millions, of users. This ranked list may also be more exhaustive than existing lists tracking usability issues. Software producers with limited resources, including volunteer-driven open source products, could thus benefit from this additional means of identifying potential usability problems.

Factors that may impact or compromise query ranking

To effectively use query analysis, it is important to understand and consider the various factors that can impact the weighting and ranking that such analysis produces. In this section, we discuss various effects that influence how often various searches are performed.

User search behaviours and query reformulation

A growing body of research (e.g., [2, 15]) examines user search behaviour. One of the practices observed is that people reformulate their queries when search results do not match their expectations or needs. As an example, a user might start with queries consisting of a few words, and then pose more detailed questions as they fail to find relevant documents in the search results [2]. As a result of this query reformulation strategy, it is conceivable that the analysis proposed in this paper artificially inflates the importance of issues for which relevant information is scarce. Reflecting on this, we note that query popularity has been observed to follow a Zipf's law distribution [26, 4]. As a result of this exponential relationship between query rank and frequency, it is unlikely that any reformulation behaviours would grossly distort the ranking of popular queries. However, the effect may be more pronounced among queries with lower search volume, suggesting a need for more work in this space.

Products with generic names

A number of products have relatively generic names (e.g., Microsoft Word, Adobe Illustrator, etc.), which can cause many irrelevant or off-topic queries to appear in query logs. A similar problem is encountered by products whose names are now synonymous with a class of operations or applications. For example, an altered digital image is often described as being "Photoshopped," regardless of which software application was used for image manipulation.

In these problematic cases, we have found our filtering techniques (e.g., "how to ___ in photoshop") are often enough to filter out the less desirable, off-topic queries.

[5] http://www.gimp.org/docs/userfaq.html
[6] http://support.mozilla.com

We also suspect that it is possible to differentiate between the uses of a word by analyzing the results that search engines return for those queries. Search engines are designed to return relevant documents, and often refine their relevance rankings by observing which pages users visit after performing searches [3]. The query-document associations recorded by search engines provide a wealth of untapped information that can further guide analysis of query logs. Use of these associations constitutes a promising area of future research.

Impact of system popularity

In the interest of preserving user privacy, query auto-completion services are unlikely to suggest queries unless those queries have been performed many times, and by many different individuals. As such, the quantity of data available for analysis by CUTS is related to the popularity of the interactive system being studied. In the extreme case, CUTS cannot be used to study systems that are not publicly available (e.g., custom software solutions, prototypes, beta software, etc.), unless one has access to internal search logs for these systems. Related to this, while CUTS is ideal for assessing usability in general, we cannot currently filter results according to particular sub-groups of users, such as novices or experts. Indeed, sub-groups in the minority are likely to be underrepresented in CUTS' output.

CONCLUSION

When faced with difficulties or questions relating to the use of interactive systems, many people routinely turn to Internet search engines as a first line of support. In this paper, we have introduced CUTS: characterizing usability through search. This process takes the name of an interactive system as input and outputs a ranked and categorized list of potential issues that users encounter with that system. These data are assembled by sampling from the query logs of top-tier Internet search engines. Importantly, the results of this process have a high degree of ecological validity, and can directly inform more formal evaluation methods by suggesting particular tasks or issues to study.

REFERENCES

1. Akers, D., Simpson, M., Jeffries, R., and Winograd, T. Undo and erase events as indicators of usability problems. In *Proc CHI '09*, ACM (New York, NY, USA, 2009), 659–668.
2. Aula, A., Khan, R. M., and Guan, Z. How does search behavior change as search becomes more difficult? In *Proc CHI '10*, ACM (New York, NY, USA, 2010), 35–44.
3. Baeza-Yates, R., and Tiberi, A. Extracting semantic relations from query logs. In *Proc KDD '07*, ACM (New York, NY, USA, 2007), 76–85.
4. Bar-Yossef, Z., and Gurevich, M. Mining search engine query logs via suggestion sampling. *Proc. VLDB Endow. 1*, 1 (2008), 54–65.
5. Barrett, R., Kandogan, E., Maglio, P. P., Haber, E. M., Takayama, L. A., and Prabaker, M. Field studies of computer system administrators: analysis of system management tools and practices. In *Proc CSCW '04*, ACM (New York, NY, USA, 2004), 388–395.
6. Brandt, J., Guo, P. J., Lewenstein, J., Dontcheva, M., and Klemmer, S. R. Two studies of opportunistic programming: interleaving web foraging, learning, and writing code. In *Proc CHI '09*, ACM (New York, NY, USA, 2009), 1589–1598.
7. Broder, A. A taxonomy of web search. *SIGIR Forum 36*, 2 (2002), 3–10.
8. Dou, Z., Song, R., and Wen, J.-R. A large-scale evaluation and analysis of personalized search strategies. In *Proc WWW '07*, ACM (New York, NY, USA, 2007), 581–590.
9. Gikas, M. Lab tests: Why Consumer Reports can't recommend the iPhone 4. *Consumer Reports* (July 2010).
10. Ginsberg, J., Mohebbi, M. H., Patel, R. S., Brammer, L., Smolinski, M. S., and Brilliant, L. Detecting influenza epidemics using search engine query data. *Nature 457* (February 2009), 1012–1014.
11. Google Corperation. Features: Google suggest. `http://www.google.com/support/websearch/bin/answer.py?hl=en\&answer=106230`, 2010.
12. Google Corperation. Google AdWords keyword tool. `https://adwords.google.com/select/KeywordToolExternal?forceLegacy=true`, 2010.
13. Google Corperation. Google suggest : FAQ. `http://labs.google.com/intl/en/suggestfaq.html`, 2010.
14. Hilbert, D. M., and Redmiles, D. F. Extracting usability information from user interface events. *ACM Computing Surveys 32*, 4 (2000), 384–421.
15. Huang, J., and Efthimiadis, E. N. Analyzing and evaluating query reformulation strategies in web search logs. In *Proc CIKM '09*, ACM (New York, NY, USA, 2009), 77–86.
16. Hurst, A., Hudson, S. E., and Mankoff, J. Dynamic detection of novice vs. skilled use without a task model. In *Proc CHI '07*, ACM (New York, NY, USA, 2007), 271–280.
17. Kellar, M., Watters, C., and Shepherd, M. A field study characterizing web-based information-seeking tasks. *J. Am. Soc. Inf. Sci. Technol. 58*, 7 (2007), 999–1018.
18. Lafreniere, B., Bunt, A., Whissell, J. S., Clarke, C. L. A., and Terry, M. Characterizing large-scale use of a direct manipulation application in the wild. In *Proc GI '10* (Toronto, Canada, 2010), 11–18.
19. Landis, J. R., and Koch, G. G. The measurement of observer agreement for categorical data. *Biometrics 33*, 1 (1977), pp. 159–174.
20. Miller, G. A. Wordnet: a lexical database for english. *Commun. ACM 38* (November 1995), 39–41.
21. Nielsen, J., and Molich, R. Heuristic evaluation of user interfaces. In *Proc CHI '90*, ACM (New York, NY, USA, 1990), 249–256.
22. Pang, B., and Lee, L. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr. 2* (January 2008), 1–135.
23. Quesenbery, W., Jarrett, C., Roddis, I., Allen, S., and Stirling, V. Search Is Now Normal Behavior. What Do We Do about That. In *UPA 2008* (Baltimore, Maryland, USA, June 2008).
24. Richardson, M. Learning about the world through long-term query logs. *ACM Trans. Web 2*, 4 (2008), 1–27.
25. Rose, D. E., and Levinson, D. Understanding user goals in web search. In *Proc WWW '04*, ACM (New York, NY, USA, 2004), 13–19.
26. Saraiva, P. C., Silva de Moura, E., Ziviani, N., Meira, W., Fonseca, R., and Riberio-Neto, B. Rank-preserving two-level caching for scalable search engines. In *Proc SIGIR '01*, ACM (New York, NY, USA, 2001), 51–58.
27. Strauss, A., and Corbin, J. *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*, 3rd edition ed. Sage Publications, 2008.
28. Zeller, T. AOL technology chief quits after data release. *The New York Times* (August 2006).