

Characterizing Search Behavior in Productivity Software

Horatiu Bota
University of Glasgow
Glasgow, Scotland, UK
h.bota.1@research.gla.ac.uk

Adam Fourney
Microsoft Research
Redmond, Washington, USA
adamfo@microsoft.com

Susan T. Dumais
Microsoft Research
Redmond, Washington, USA
sdumais@microsoft.com

Tomasz L. Religa
Microsoft Corporation
Redmond, Washington, USA
toreli@microsoft.com

Robert Rounthwaite
Microsoft Corporation
Redmond, Washington, USA
robertro@microsoft.com

ABSTRACT

Complex software applications expose hundreds of commands to users through intricate menu hierarchies. One of the most popular productivity software suites, Microsoft Office, has recently developed functionality that allows users to issue free-form text queries to a search system to quickly find commands they want to execute, retrieve help documentation or access web results in a unified interface.

In this paper, we analyze millions of search sessions originating from within Microsoft Office applications, collected over one month of activity, in an effort to characterize search behavior in productivity software. Our research brings together previous efforts in analyzing command usage in large-scale applications and efforts in understanding search behavior in environments other than the web. Our findings show that users engage primarily in command search, and that re-accessing commands through search is a frequent behavior. Our work represents the first large-scale analysis of search over command spaces and is an important first step in understanding how search systems integrated with productivity software can be successfully developed.

KEYWORDS

Environment specific retrieval, command search

ACM Reference Format:

Horatiu Bota, Adam Fourney, Susan T. Dumais, Tomasz L. Religa, and Robert Rounthwaite. 2018. Characterizing Search Behavior in Productivity Software. In *CHIIR '18: 2018 Conference on Human Information Interaction & Retrieval*, March 11–15, 2018, New Brunswick, NJ, USA. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3176349.3176395>

1 INTRODUCTION

In a 2007 column, Don Norman argued that graphical user interfaces would struggle to scale under the increasing complexity of software applications, and that scaling could be achieved by embracing rich

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

CHIIR '18, March 11–15, 2018, New Brunswick, NJ, USA

© 2018 Association for Computing Machinery.

ACM ISBN 978-1-4503-4925-3/18/03...\$15.00

<https://doi.org/10.1145/3176349.3176395>

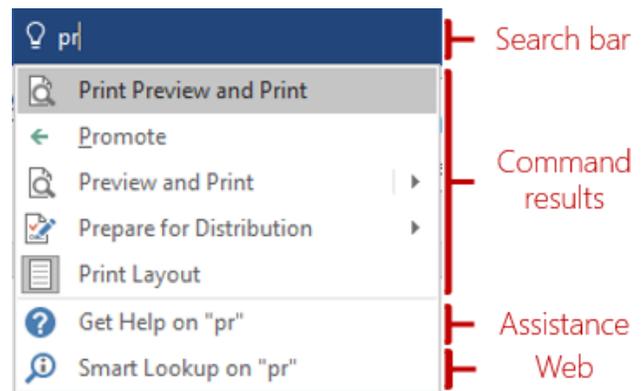


Figure 1: Searching for “print” in Microsoft Word.

search experiences [19]. This transformation was predicated on search interfaces gaining, or strengthening, an ability to directly execute actions in addition to retrieving documents. In the intervening ten years, rich search experiences, combining document retrieval with action execution, are now available across a variety of surfaces, including: web search engines, operating systems, and feature-rich applications. For example, on both Bing.com and Google.com, the query [*stopwatch*] both summons an interactive stopwatch, and retrieves documents about this timepiece. Likewise, on both the Windows 10 and macOS operating systems, the query [*bluetooth*] both accesses the device’s Bluetooth settings, and optionally retrieves web pages about Bluetooth technology. Finally, users of both Microsoft PowerPoint and Adobe Photoshop can issue the query [*crop*] to either execute the application’s crop command, or to retrieve documentation about this tool.

Core to these experiences, and to Norman’s vision, is that search systems effectively respond to queries where the searcher’s intention is to perform some action (web-mediated, or otherwise). Such search intents were first observed in the query logs of the AltaVista search engine [4, 21], and have varyingly come to be known as *transactional queries* [4], or *resource interaction* queries [21]. Historically, transactional queries represent a minority of all web searches.

More recently, researchers have begun characterizing search interactions occurring via the virtual assistants that are integrated into modern operating systems [7, 15]. In this environment, log analysis has shown that roughly half of all queries result in the

direct execution of an action (e.g., setting an alarm) [15]. As such, virtual assistants represent a mid-point along a spectrum of search systems that support direct execution of transactional queries.

In this paper, we provide the first large-scale log-based characterization of search interactions within the context of productivity software, namely Microsoft Office (Figure 1). Microsoft Office, and similar in-application search experiences (e.g., [3, 16]), represent an extreme point along the aforementioned spectrum – an understudied search environment where most queries are intended to execute an action. To investigate this environment, we analyze the queries of a million Microsoft Office users, engaging in millions of search sessions over a one month period in 2017. With these data, we address the following research goals:

- (RG1) Describing search behavior:** In particular, we answer the following questions with respect to search in productivity software: **(1.1)** How is search distributed across queries, accessed commands and users? **(1.2)** What types of search do users engage in and how frequently? **(1.3)** How do users engage in different types of search activity? Specifically, we want to characterize how users engage in command and control search in contrast to informational search.
- (RG2) Describing search abandonment:** Users frequently abandon their sessions without clicking on results returned by the search system. We set out to contrast abandoned search to searches in which commands are executed, and we highlight properties of search results and rankings that potentially influence abandonment.
- (RG3) Describing re-ranking methods:** The use of behavioral signals in improving search results ranking is widespread in modern web search [2, 25]. Therefore, we explore methods of using historical user interaction data and simple user engagement metrics in re-ranking command lists, with the intent of improving ranking quality in rich application search.

The rest of this paper is structured as follows: We review prior research, then describe the search experience that is integrated into many Microsoft Office products. We describe the specific instrumentation data we analyzed, and present results that address our research objectives. We conclude with a discussion of the implications of our findings for the development of in-application search experiences.

2 RELATED WORK

This work builds on three distinct threads of prior research which we review here. We begin by discussing transactional queries in the context of web search, and their generalizations to other domains. We then review research that characterizes command invocations in feature-rich software. Finally, we review past and ongoing efforts to integrate rich search experiences into feature-rich software.

2.1 Transactional Queries

Transactional queries were first characterized by Andrei Broder as web searches whose “*intent is to perform some web-mediated activity*” [4]. Broder interpreted this definition quite broadly, including queries intended to initiate e-commerce, as well as those intended to access phone directories, weather services, and other

on-line databases. Upon examining 400 searches performed on the AltaVista web search engine, Broder reported that 30% of all queries met this definition. Other work has consistently placed transactional queries in the minority of all web searches, though the precise proportion varies by search engine [20], and by query topic [13].

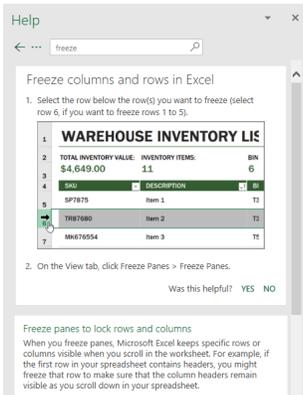
In follow-up work [21], Rose and Levinson proposed a refinement to Broder’s taxonomy, subdividing transactional queries into four subcategories. Relevant to this work is the subcategory “*interact*”, whose queries include those where the searcher’s “*goal is to interact with a resource*” (e.g., a weather service, or a unit converter). Again using AltaVista search logs, Rose and Levinson reported that between 4.6% - 6.0% of all queries met this narrower definition.

Web search engines have since moved to directly support Rose and Levinson’s *interact* queries by presenting tailored experiences directly on search engine results pages. In 2011, Chilton and Teevan reported that the Bing.com search engine supported nearly 100 such experiences [5]. Separately, Stamou and Efthimiadis studied donated search logs, and reported that 27% of web searches were intended to trigger these types of quick answers [22]. These trends prompted Don Norman to observe that “*more and more, we type commands (into search engines), not search items*”, and to further describe search as the “*modern command language*” [19].

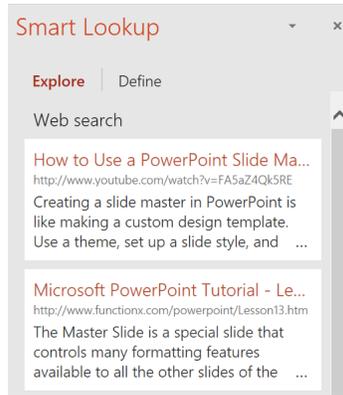
Importantly, Norman’s observations were not confined to web search: desktop search, email search, and an early version of Microsoft Office search are explicitly mentioned in [19]. Broder’s definition of transactional queries naturally generalizes to these environments by simply lifting the constraint that the intended activities be web-mediated. Through this lens we consider work by Jiang et al., who analyzed queries posed to the Cortana virtual assistant [15], the successor of desktop search on the Windows 10 operating system. Jiang et al. reported that roughly half (47%) of all queries result in the direct execution of an action (e.g., setting an alarm, or playing a song) and thus meet the generalized definition of transactional queries. Moreover, this figure may underestimate the prevalence of transactional queries, as it excludes utterances that have a transactional intent, but which fall outside the capabilities of the virtual assistant [7, 24]. In this paper, we extend this generalization of transactional queries, and Broder’s query taxonomy, by characterizing search behaviors in Microsoft Office – one of several well-known applications to recently allow commands to be triggered through in-application search.

2.2 Command Invocations in Software

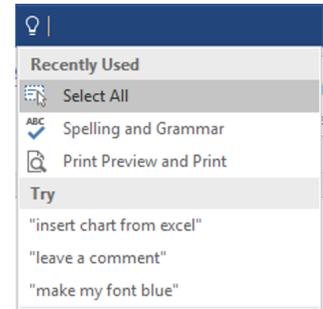
As noted above, Norman characterized search as the “*modern command language*” [19], and, in doing so, drew a direct comparison to a prior generation of command-line interfaces. Command-line systems have themselves been the subject of extensive study [10–12], and there are several relevant findings that we report here. First, as with web search [14], command invocations are known to follow an inverse power law distribution, with the most common commands occurring exponentially more often than less popular commands [11, 12]. Second, though each user will tend to frequent a small set of commands, there is little overlap in the command vocabularies between users [11]. Finally, even when users have the same commands or intentions in mind, they will often use different words or terminology to describe these intentions to the system,



(a) Interface side panel displaying help documents retrieved by the search system for the query “freeze” in Excel.



(b) Interface side panel displaying web search results for the query “slide master tutorial” in PowerPoint.



(c) Search interface displaying three of the most recently used commands and query suggestions, as shown before query input in Word.

Figure 2: Example (a) help search results, (b) web search results and (c) no-query interactions.

posing problems for command-line interfaces and search systems alike [10]. In this paper, we touch on each of these points, as they relate to commands issued in Microsoft Office via a search interface.

Graphical user interfaces (GUIs) are also command languages – albeit visual rather than textual. To this end, research has found that the aforementioned properties of command-line interfaces also apply to GUIs [17], and to the web search queries that users issue to support their use of those systems [6]. This has spawned work on two fronts: First, numerous command recommendation systems have been developed to help with command discovery, and to broaden users’ command vocabularies [9, 18, 26]. Second, to address the vocabulary problem, researchers have proposed integrating search experiences into graphical interfaces, including Microsoft Office, so that a user’s free-text queries can be more flexibly matched to system commands. We describe these search systems next.

2.3 In-Application Search

Search is becoming a popular method of accessing commands, settings, and other functionality in modern feature-rich applications. In addition to Microsoft Office, Adobe’s creative products now feature a universal in-application search experience [16]. Likewise, the macOS operating system has long offered a search field in the system-wide help menu, providing its users with a means of searching and executing commands found in the top few levels of any application’s menuing system [3]. Similarly, search is the primary means of accessing settings in the Chrome web browser, and on all modern operating systems and platforms (Windows, macOS, iOS, Android, etc.)

Embedded in each of these in-application experiences is the need to return relevant commands, settings, and documents in response to user queries. In GUIs, there is often very little text associated with each command or setting, aggravating the aforementioned vocabulary problem, and limiting the effectiveness of simple ranking algorithms that depend on term or character overlap [6]. In the literature, several systems address this retrieval problem. For example, Fournay et al. [8] leveraged web search queries, together with on-line software tutorials, to better map keywords to system

commands. Likewise, Adar et al., improved command search by learning a command-to-word embedding from millions of on-line tutorials [1]. In this work, we demonstrate how behavioral data can be used to improve command ranking.

In summary, our work extends prior research by characterizing transactional queries and command invocations in the context of a widely deployed suite of software applications. As such, our findings are derived from a comprehensive analysis of in-application query logs, and our work is the largest such study of its kind. We begin by providing a review of search functionality in Microsoft Office.

3 SEARCH IN MICROSOFT OFFICE

Microsoft Office is a collection of productivity software developed for knowledge workers, with millions of active users each month. Recent versions of Office¹ include a unified search interface that allows users to issue free-form text queries to either find and quickly access commands, search through Office help documentation or retrieve Web results directly in the application interface. The unified search functionality is provided through a search box that prompts users to “Tell me what you want to do”, located above the application command menu (the *ribbon*).

In this paper, an in-application “search” starts when the user activates the query input area (the *search box*), and ends when the user clicks one of the results returned by the system, or deactivates query input by clicking outside the search area. This section provides an overview of search in Microsoft Office, and describes the types of results returned by the system.

3.1 Result Types

Three types of results are accessible through Office search: commands, help documents and web results. We describe each below.

Command Results. Figure 1 shows an example of a Microsoft Word user issuing the query “pr” to the search system. Up to five command results are displayed immediately underneath the search box, and are updated continuously as users type. Results are ranked

¹Starting with Microsoft Office 2016.

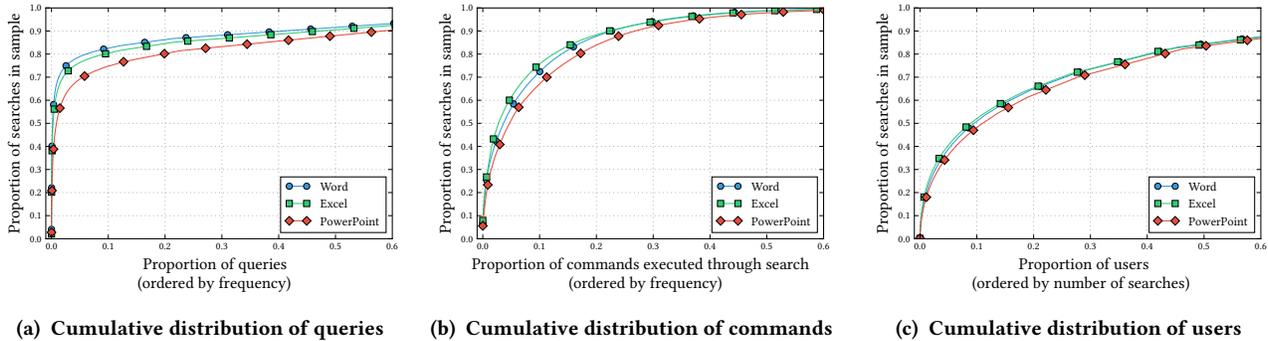


Figure 3: Distribution of queries and commands executed via search functionality

by their relevance to the user query, and need not be lexically similar to command names.

Command results returned by the search system can be of three sub-types: (i) *action* commands that initiate a process (for example, the “Print” command opens an additional dialog that users need to navigate in order to complete their task); (ii) *flag* commands which toggle a binary feature (such as the “Bold” command) without requiring any additional action from the user; (iii) *menu* commands which, on user click, open an additional side menu containing related sub-commands that the user can browse and click – in Figure 1, menu commands are displayed with a chevron icon on the right of their label. In addition, commands returned to users are disabled (greyed-out, not clickable) if the active system state is incompatible with their execution (e.g., the *Group* command is greyed-out when no items are selected for grouping). We investigate the role these command types serve in influencing search behavior in §5.5.

Help Results. Office search allows users to quickly retrieve help documentation using free-form text queries. Unlike commands, help results are not displayed directly beneath the search bar, but in a side panel where the user can refine their query and navigate returned documents. To activate the help search side panel, users need to click on the “Get Help” button shown below the command results list, as shown in Figure 1. Figure 2a shows an example of the help search side panel for the query “freeze” in Microsoft Excel.

Web Results. Users can also retrieve and display web documents directly in the application interface through Office search. The interaction pattern is similar to accessing help documents, in that web results are displayed in a side panel, where the user can explore returned documents. The web search side panel is activated by clicking on the “Smart Lookup” button below the command results list. Figure 2b shows an example of web documents returned for the query “slide master tutorial” in Microsoft PowerPoint.

3.2 Zero-query Interactions

After activating the search box, but before any query input, the search results area is populated with query suggestions (Figure 2c, bottom), and up to 5 of the most recent commands previously found and executed through search (Figure 2c, top).

3.3 Abandoned Search

Search sessions are regularly abandoned, with the user deactivating the search box (e.g., by clicking outside the search interface) without interacting with any of the results returned by the system. We explore abandonment in the context of in-application search in section 5.5 in more detail, but we highlight here the two types of search abandonment we observed in our analysis: (i) **abandoned (query)**: search sessions in which the user explicitly types a query in the search box and decides not to interact with any of the results returned by the system; and (ii) **abandoned (zero-query)**: search sessions in which the user activates the search area, but does not issue a query or interact with the search interface (e.g., recently used commands) before deactivating the search menu. We highlight these two types of abandonment (*with* and *without* query) because they are driven by different underlying user expectations with regard to system functionality.

4 DATA

To understand search in productivity software and address our specific research goals, we make use of Office instrumentation logs, which include: (i) *information about the user*: anonymized user identifier; (ii) *information regarding application context*: application version number, users’ system locale and language settings; (iii) *information about search sessions*: timestamps, user-typed queries, result rankings shown to user, and clicks.

From the instrumentation logs, we extracted English-language queries performed by Microsoft Office users who reside in the United States. Although search functionality is available in most Office products², we focus our analysis on three of the most popular applications in the suite: Word, Excel and PowerPoint. In addition, our data was obtained from a single Office build version, to ensure consistency of features available to users over incremental releases of Microsoft Office. Our data covers search events over a four week period, from the 29th of May to the 25th of June 2017. Finally, from these data, we randomly sampled one million users and their millions of queries.

5 RESULTS

We begin this section by describing the properties of query, command and user distributions in Office search (§5.1). We structure

²Version 16 and higher.

our results by providing a comparison between different types of in-application search (§5.2 – §5.4), followed by a closer look at characteristics of abandoned search (§5.5); lastly, we describe ways of using behavioral data to improve command ranking quality (§5.6).

5.1 Query, Command and User Distributions

Similar to Web search [14], the distribution of queries observed in Office search is long-tailed, with a small number of query strings accounting for a large proportion of the search events, across applications. Figure 3a shows the cumulative distribution of distinct query strings observed in our sample, ordered by frequency. Roughly 10% of query strings account for 80% of searches in our sample. This trend is consistent across Office applications, suggesting that productivity search is similar across the three applications we studied. Figure 3b shows the cumulative distribution of *commands executed* via the search functionality. Similarly, the top 10% most popular commands executed via search account for roughly 70% of all search-issued commands – although the shapes of the two distributions suggest a heavier tailed query distribution. Finally, Figure 3c displays the cumulative distribution of searches across users, showing that the most active 10% of users account for roughly 50% of the searches in our sample.

5.2 Engagement Across Result Types

When users issue search queries in Office, the results may include commands, a link to Office help, or a link to Web search results. Among these three, command execution is the most likely outcome; command results are clicked 6.4 times as often as help documentation, and 32 times as often as web search. This indicates that most searches in Office are transactional, and are used to access – or re-access – commands.

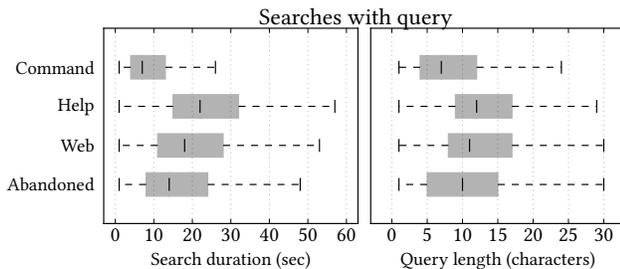


Figure 4: Search session durations and query length.

Figure 4 shows the distribution of search session duration and query length across types of search outcomes. Search session duration is the duration in seconds between search bar activation (e.g., by a user click on the search bar) and search bar deactivation (e.g., by a user clicking on one of the results), and query length is the number of characters in the query at search bar deactivation.

The median duration of search sessions that terminate with a command execution is 7 seconds, whereas the median duration for search sessions that terminate with the activation of help or web search panels is 22 and 17 seconds respectively. Similarly, command search queries have a median length of 6 characters, whereas help or web search queries have median lengths of 12 and 11 characters respectively. These differences highlight that informational

intent (i.e., the combination of help and web results) is typically expressed through longer queries and takes longer to formulate than command search. However, there is only a weak positive correlation between session duration and query length (*Pearson’s* $r = 0.39, p < 0.001$), suggesting that the typing effort is not the only factor behind the differences we observed. We now contrast command search to informational search in more detail.

5.3 Command (Transactional) Search

Given that command access is the most frequent search intent in Microsoft Office, we take a closer look at this activity. Table 1 shows the top 10 most frequent queries for each of the three applications in our dataset. Though each of these queries was explicitly typed by the searcher, user behavior is influenced by queries *previously* suggested via the no-query experience (as described in §3.2). These *primed queries* are over-represented in our dataset, and are denoted with †. We note that *primed queries* are abandoned at a much higher rate than other queries (%Abandoned), perhaps indicating users are exploring the system, rather than satisfying actual search needs. We discuss abandonment in detail in §5.5.

It is interesting to note that previous work on command execution in large-scale applications has shown a strong overlap among users in their access of the most popular commands, and low overlap for commands in the torso or tail of the corresponding command distribution [11, 17]. In contrast, our analysis of in-application search shows that there is low overlap in queries across users (%Users column; similarly reported with respect to command usage in [17]) – even with respect to the top searches – even though queries often directly match the names of commands. One possible interpretation is that users are leveraging in-application search to access idiosyncratic commands in their command vocabularies.

In addition to frequency, we report the proportion of *users* who issued each query (User Rank). This metric provides an indication of how popular a given query is across users. We highlight two examples: the query “undo” for Word, although it is the 5th most popular query by number of issues, it is the 15th most popular by the number of users who have issued the query at least once. Similarly, the query “group”, for PowerPoint, ranks 10th most popular by frequency of issues, but 17th most popular by number of users who have submitted it to the search system at least once. This shows that a smaller number of searchers make frequent use of these queries. We discuss these re-access and re-finding patterns in §5.4.

Finally, queries used to access commands vary widely. Table 2 presents a different perspective on command access, by showing the three most frequent commands executed through search, along with the five most popular queries used to retrieve these commands, and their relative frequency. There is less variation in the queries used to retrieve commands such as “Print” in Word or “Crop” in PowerPoint, where the command name also maps directly to the query used most frequently to retrieve it – 80% of retrievals of the “Crop” command being achieved via the query “crop”. On the other hand, Table 2 also shows examples of commands that are retrieved by a more varied set of queries, such as “Line Spacing” in Word or “Orientation” in PowerPoint. Because these commands require parameters (e.g., the amount of spacing between lines), users, in addition to using the command name as a query, may

Query	Most Popular Command	% Search Volume	% App Sessions	% Users	% Abandoned	% Requery	# Query Rank	# App Rank	# Users Rank
Microsoft Word									
print†	PrintDefault	4.06%	5.95%	7.21%	71.99%	27.20%	1	1	1
write an essay†	Researcher	1.48%	2.36%	3.14%	99.48%	12.98%	2	2	2
word	WordCount	1.20%	1.12%	1.22%	2.86%	58.14%	3	5	6
spell	Proofing	1.00%	1.54%	1.77%	2.49%	27.78%	4	3	4
undo	Undo	0.88%	0.81%	0.85%	3.26%	60.22%	5	12	15
share my document†	Collaborate	0.84%	1.43%	1.96%	99.90%	4.28%	6	4	3
find	Find	0.80%	1.12%	1.17%	6.61%	40.05%	7	6	7
water	Watermark	0.80%	1.05%	1.13%	5.96%	41.69%	8	8	8
sp	Proofing	0.76%	1.10%	1.23%	2.52%	33.73%	9	7	5
page	PageNum	0.60%	0.81%	0.97%	12.11%	32.90%	10	11	12
Microsoft Excel									
header	HeaderAndFooter	2.09%	3.04%	3.32%	6.88%	39.17%	1	1	1
free	FreezePanes	1.70%	1.96%	2.11%	8.25%	52.57%	2	3	3
print	PrintDefault	1.44%	2.19%	2.53%	16.21%	32.90%	3	2	2
sort	Sort	1.18%	1.29%	1.44%	11.85%	53.38%	4	8	7
find	Find	1.07%	1.35%	1.27%	7.35%	54.82%	5	5	9
insert	InsertSheetRows	1.06%	1.14%	1.24%	23.88%	55.58%	6	10	10
freeze the top row†	FreezePanes	0.93%	1.43%	1.92%	99.78%	20.93%	7	4	4
insert row	InsertSheetRows	0.85%	0.93%	1.06%	8.62%	52.43%	8	11	12
freeze	FreezePanes	0.80%	1.16%	1.41%	13.41%	32.94%	9	9	8
insert a table†	InsertList	0.79%	1.34%	1.79%	99.77%	13.77%	10	7	6
Microsoft PowerPoint									
crop	Crop	2.76%	2.33%	2.47%	20.55%	64.18%	1	2	3
start presentation†	StartSlideshow	2.17%	3.76%	4.57%	99.55%	15.55%	2	1	1
change slide background†	FormatBackground	1.31%	2.19%	2.72%	99.42%	16.98%	3	3	2
portrait	Orientation	1.19%	1.94%	2.41%	7.27%	18.72%	4	4	4
de	DesignerPane	0.94%	1.04%	1.12%	4.15%	52.11%	5	6	7
master	SlideMaster	0.84%	1.02%	0.97%	1.61%	53.58%	6	7	11
des	DesignerPane	0.71%	0.82%	0.88%	2.59%	50.38%	7	12	13
design	DesignerPane	0.67%	0.90%	0.97%	6.05%	41.81%	8	9	10
change layout of slide†	MasterStyle	0.64%	1.09%	1.37%	99.66%	14.24%	9	5	5
group	Group	0.64%	0.69%	0.77%	35.03%	51.79%	10	15	17

Table 1: Head queries per application. Priming queries are marked with the (†) symbol.

Microsoft Word			Microsoft Excel			Microsoft PowerPoint		
Command Name	Query		Command Name	Query		Command Name	Query	
Proofing	spell	28.07%	FreezePanes	free	39.69%	Orientation	portrait	24.38%
	sp	18.51%		freeze	14.87%		orientation	6.65%
	spe	15.27%		fre	13.30%		landscape	5.02%
	spelling	6.35%		fr	11.82%		change to portrait	3.94%
	spell check	6.25%		freez	4.34%		change orientation	3.20%
WordCount	word	42.96%	HeaderAndFooter	header	56.65%	DesignerPane	des	23.29%
	word count	20.13%		footer	13.71%		design	19.80%
	wor	12.15%		head	9.03%		de	16.39%
	wo	4.86%		foo	3.71%		desi	8.33%
	character count	4.40%		foot	3.23%		d	6.10%
LineSpacing	line	14.94%	InsertSheetRows	insert row	33.72%	Crop	crop	81.16%
	spacing	6.80%		insert	27.15%		cr	12.42%
	single space	6.61%		insert a row	5.82%		crop picture	0.60%
	double space	6.19%		inser	4.67%		finish crop	0.39%
	double	4.78%		insert rows	3.64%		crop image	0.34%

Table 2: Most frequently used queries for executing top three most frequent commands executed through search.

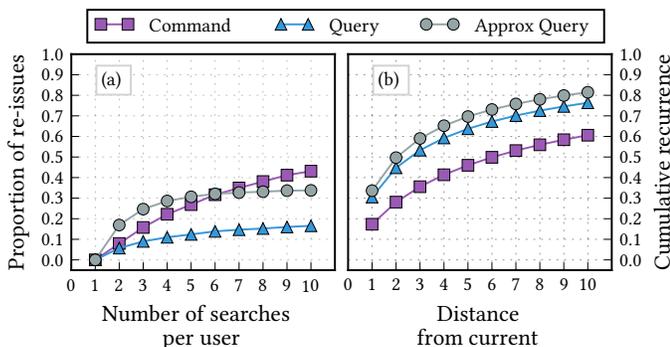


Figure 5: Search recurrence.

also specify parameters of the command as part of their queries. This behavior is informative for the development of search systems for productivity software, where command outcomes, rather than command descriptions are used for search (e.g., users issuing the query “blue text” rather than “change font color”).

5.4 Re-finding

Re-finding is an important aspect of search. As in Web search, where up to 39% of searches are driven by accessing previously retrieved resources [23], re-finding is frequent in Microsoft Office search. Not only do searchers frequently make use of the recently used commands list, they repeatedly issue the same queries. Table 1 shows, for a given query, the proportion of searches for which users had previously issued the same query within our sampling period (% Requery). There is variation among users and queries with respect to their re-querying rate, but on average, the priming queries are the least re-issued by users (and most likely to be abandoned).

The aforementioned view of re-querying is muddled by the fact that people varied in their use of the Office applications in general, and of the in-application search experience in particular (i.e. people with more sessions or queries had more opportunities for re-accesses). Figure 5(a) shows the proportion of re-accesses, but partitions users into cohorts based on the total number of queries they issued during the month we studied. We distinguish three separate re-access types: re-executing a command found through search (“Command”), re-issuing a query to the search system (“Query”) or re-issuing a prefix or an extension of a previously issued query – for instance, the queries “pr” and “prin” – (“ApproxQuery”). For each of these types of search engagement, Figure 5(a) shows the distribution of re-access over repeated use of the search system. It is interesting to note that, for users with three distinct searches in our sample, the proportion of repeated queries is 9%, but the proportion of repeated commands is 17%, and that of approximately repeated queries is 26%. This suggests that users learn approximately equivalent queries (e.g., a set of prefixes) for accessing the same commands.

We also examine the distance (i.e. the number of search activations) between re-access instances. In this view, we include only search sessions that are indeed instances of re-access for a given user. As seen in Figure 5(b), 30% to 35% of repeated queries are identical to, or are approximations of, the previous query issued by the user. Moreover, a history window of 5 queries is sufficient to

explain 70% of approximate query re-issues. Conversely, command re-access is distributed more evenly across the five most recently executed commands, primarily because the recently used command list provides easy access to these commands. Even so, almost 20% of the commands that are re-executed through search are repeats of the most recently executed command, and the top five most recent commands account for roughly 45% of command repeats. We now take a closer look at factors influencing search abandonment.

5.5 Search Abandonment

Like in web search [22], searches in Office are frequently abandoned. Abandonment can occur when a user inputs a query and elects not to click a result, or when a user activates the search bar without issuing a query and elects not to click on a recently used command or suggestion. In either case, a user may fail to click on a result either because the intended command is not listed, or, importantly, because it is present but the active system state is incompatible with its execution (i.e. the command is greyed-out). For example, in PowerPoint an object must be selected for the *Crop* and *Group* commands to be enabled, and thus the corresponding queries “crop” and “group” have high abandonment rates compared to similar queries (20.55% and 35.03% respectively, as seen in Table 1). We return to these system state errors, and their role in search abandonment, later.

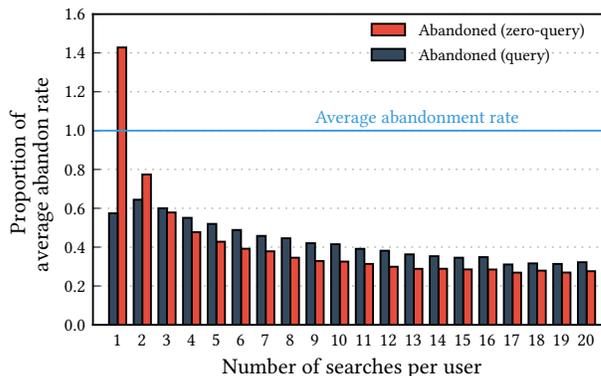


Figure 6: Distribution of search abandonment over user groups partitioned by the number of search instances.

In our data, abandonment is twice as common in the zero-query condition than in cases when users type at least one character into the search box. Users just exploring the search bar as a novelty item or mistakenly activating the search bar through click or keyboard shortcut might partially explain abandonment without query input. Figure 6 shows the distribution of search outcomes over groups of users with repeated search access, sorted from users who have used search functionality a single time (left) to users who have used search functionality exactly 20 times (right) over the time period we observed in our sample. It is interesting to note that the relative proportion of abandoned search decreases with repeated system use, as users either populate their recently used command list with relevant or frequently accessed commands, or they learn and adapt to the capabilities of the search system. We now take a closer look at search abandonment after query input.

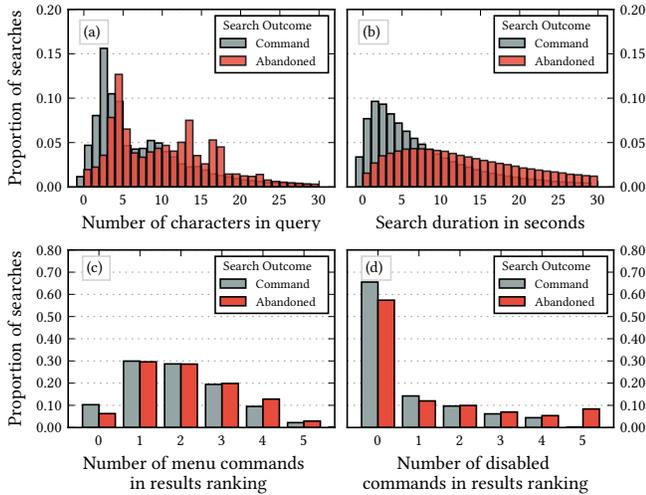


Figure 7: Differences between command search and abandoned (query) search.

5.5.1 Differences between command and abandoned search. Understanding the characteristics of abandoned search in contrast to successful command search is critical to developing rich search functionality that meet users’ expectations and needs. Figure 7 shows differences between command and abandoned search with respect to query length (7a), search session duration (7b), number of menu results shown in the search ranking, (7c) and number of disabled results shown in the search ranking (7d). The first row in Figure 7 shows differences in search session properties, such as query length and duration. Overall, for abandoned search sessions we observe longer queries (median query length 14 characters) and longer search duration (median duration 14 seconds) than for command search. This suggests that, in the case of abandoned sessions, users express more complex needs to the system, and that search properties, such as query length and duration, can be used to identify searches in which users are struggling and perhaps pre-empt abandonment by offering additional support.

The second row of Figure 7 shows differences based on the number of menu type results shown in the ranked list of commands, and the number of disabled results in the ranking. In both cases, abandoned search sessions typically display more menu type results and more disabled items – almost 8% of abandoned searches do not lead to a command execution simply because all items returned to the user are disabled due to system state (e.g., searching for “crop” when no items that can be cropped are selected). It is worth noting that showing a disabled command is arguably better than not showing it at all, which might leave the user uncertain about whether the command exists, or if the system understood the intent of the query. Overall, these findings suggest that the types of commands returned to the user have an effect on search behavior and we further investigate the role of result type in search abandonment.

5.5.2 Command types and search abandonment. Command results returned by the search system in Microsoft Office are not uniform with respect to their user interaction patterns. Section 3.1

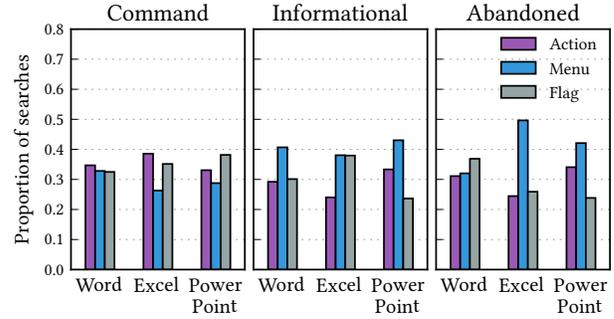


Figure 8: Distribution of search outcomes over command types, controlling for how often each command type is displayed at top rank.

reviews the different types of command results returned to users: *action*, *flag* or *menu* commands. Given that, overall, a majority of user clicks on command results are issued on the top-ranked result, in this section we inspect the effects of top-ranked command type on user search behavior.

The type of command displayed at top rank in the search results returned varies with both user intent and with the distribution of command types available in each of the applications we explored. Table 3 shows the distribution of command types shown at top rank. Although there is some variation across applications, the most common types of results returned at top rank are *action* commands and *menu* commands.

	Action	Flag	Menu
Word	49.88%	10.88%	39.24%
Excel	44.45%	13.16%	42.40%
PowerPoint	34.89%	21.29%	43.82%

Table 3: Distribution of command types displayed at top-rank.

Figure 8 shows the distribution of search outcomes over command types, normalized by the relative frequency of command types at the top rank for each application. Given this normalization, in a uniform setting, each command type (i.e. bar color) should account for $1/3^{\text{rd}}$ of the search outcomes. However, our results show that menu commands are more frequent at top rank in the case of abandoned and informational search. This finding suggests that the menu command type might influence users decision to abandon their searches or seek additional support. One hypothesis that may explain this behavior is that relevant command results may not be readily apparent to the user until they take the additional action of opening the sub-menu, and that not all users will take this additional step. Another hypothesis is that menu commands reflect more complex or obscure tasks, which may pose challenges in both query formulation and task execution. As such, it is not the menu result that leads to abandonment, but rather the more complex task the user is engaged in. Even so, our findings suggests that extracting sub-commands from deeper menu hierarchies and displaying sub-commands directly in the search results list might benefit users trying to complete multi-step actions through search.

5.6 Re-ranking Results

In this section, we report on the application of command re-ranking strategies informed by behavioral data. We evaluate our re-ranking strategies in an off-line setting, using log data, by interpreting clicks as positive relevance labels. There are two aspects of result re-ranking that we discuss here: re-ranking *strategy*, by which we refer to the reordering of commands returned to users based on simple metrics; and re-ranking *selection*, by which we refer to the process of selecting *queries* that might benefit from results re-ranking. Informed by our analysis of user engagement with Office search, we consider the following strategies:

Result type re-ranking. Based on our observation that searches in which *menu* commands are top-ranked seem to be abandoned at a higher rate, we re-rank results by explicitly placing menu commands below *action* or *flag* items in the ranking.

Per-query command click-through rate. User engagement with the search system allows us to observe command execution frequency and click-through rate for a given query. For a query and a list of results relevant to the query, we re-rank commands in descending order of their per-query click-through rate.

Overall command click-through rate. Similar to the previous strategy, we can compute global command click-through rates for individual commands, as a measure of how useful a command is overall. Given a list of results, we re-rank commands in descending order of their global click-through rate.

To evaluate the strategies outlined above, we split our data into training and test folds (each fold covering a non-overlapping period of two weeks). We use training data for two purposes: firstly, to compute overall and per-query command click-through rates; secondly, we use the training fold to select which *queries* (and corresponding rankings) potentially benefit from re-ranking. Selecting which result lists to re-rank is not trivial, given that more than 70% of searches retrieved optimal rankings (i.e. clicked command returned at top rank), and as such, re-ranking these searches would likely deteriorate the quality of their results. Thus, we compare two different methods to select instances of search in which to deploy our re-ranking strategies.

Historical query selection. We use training data to select *queries* that generated clicks on lower ranked results, on average, and we apply our re-ranking strategies only to those queries, as observed in our test data. Using this filtering method, we select 16.34% of the searches in our test fold for re-ranking.

Oracle query selection. Given that our logs contain the clicked result rank, we can identify all searches in which clicks were not issued on the top-ranked result, and apply re-ranking strategies to these searches. This selection method is not feasible in a real-world setting, where click rank is not known beforehand, but is an informative baseline with respect to the effectiveness of our re-ranking strategies. Using this filtering method, we select 29.46% of the searches in our test fold for re-ranking.

Table 4 shows the distribution of clicks over ranks for our re-ranking strategies. The existing system ranker returns the clicked result at top rank in more than 70% of searches in our test data. Even so, all re-ranking strategies we explored increase the proportion of clicks at top-rank. Simply placing *menu* type results lower in the ranking increases the proportion of top-rank clicks by 0.6% to

Selection	Strategy	Clicked result rank				
		1	2	3	4	5
	No re-ranking	70.54%	15.93%	7.18%	3.85%	2.50%
Historical	Result type	71.17%	12.47%	7.21%	5.28%	3.87%
	Overall click-through	74.66%	12.52%	6.47%	3.83%	2.51%
	Per-query click-through	77.93%	12.29%	5.32%	2.68%	1.78%
Oracle	Result type	76.77%	7.48%	5.76%	5.63%	4.36%
	Overall click-through	82.65%	7.25%	5.08%	3.20%	1.82%
	Per-query click-through	83.18%	9.47%	4.24%	1.99%	1.12%

Table 4: Distribution of clicks over ranking positions.

6%, in historical and oracle re-ranking query selection, respectively. Together with our analysis of command types and their role in abandoned search, our findings suggest that placing menu items lower in the ranking does not deteriorate the quality of results returned to the user, and might even prevent abandonment. We hypothesize that users willing to navigate menu hierarchies to locate their intended command will do so even when the menu result – which collapses the relevant menu hierarchy – is at a lower rank; the converse might not be true, with users looking for quick access to an action (or flag) command perhaps being less inclined to review the sub-menu contents of top-ranked menu results in order to locate their intended command, and maybe more likely to abandon their search. We leave testing this hypothesis for future work.

Overall, the most effective re-ranking metric we explored is *per-query command click-through rate*. Re-ranking based on this metric increased top-rank clicks by 7% to 13% over historical and oracle query selection methods, respectively. Even though integrating behavioral signals into search results ranking is widespread in search algorithms for the web [2, 25], their application to in-application command search is under-explored and our work provides an overview of simple re-ranking strategies based on user interaction data as a first step towards integrating behavioral signals into command and control search.

6 CONCLUSIONS

Rich search experiences have become available across a variety of surfaces: from the web, to personal information repositories, feature-rich applications and operating systems. These novel search experiences transform both the way users engage with complex applications, and their expectations of search systems in general.

Our study provides a characterization of user behavior in an under-studied area of information access: search in productivity software. Millions of users actively engage with productivity software each month. As search interactions become integrated into their workflows, understanding search behavior is necessary for developing systems that can effectively respond to users’ queries.

Users primarily engage in finding commands through the search interface available in Microsoft Office (**RG1.2**). The distributional properties of observed queries are similar to other search domains, in that a large proportion of search volume is generated by a small proportion of queries (**RG1.1**). However, our results show that, unlike command access, there is low overlap in queries across users – even for frequently observed query strings. One possible interpretation is that users are leveraging in-application search to access idiosyncratic commands in their command vocabularies. We also

show that queries used to access commands vary widely, and that action outcomes (e.g., “*portrait*”) rather than command names (e.g., “*change orientation*”) are commonly used to retrieve parameterized commands (**RG1.3**). Users frequently engage in command re-access through search, not only by using the zero-query interface available in Microsoft Office, but by explicitly re-issuing queries to the system. For frequent users of search, on average, more than 26% of queries are approximate repetitions of a previously issued query. Moreover, up to 70% of repeated queries occur within a window of five most recent user queries. Together with our characterization of observed queries, our findings regarding re-access are informative for the development of command retrieval systems (**RG1**).

Identifying instances of search in which users are unable to access intended commands is a crucial element of improving search quality. Our work on abandonment shows that successful command search and abandoned searches vary with respect to their behavioral properties, such as query length and search duration, and with respect to result ranking characteristics, such as the number of disabled or menu commands shown in the results list (**RG2**). Furthermore, we indicate that different command types influence search outcomes, and show that menu commands are prevalent at top rank in abandoned searches. Our findings suggest that ranking sub-commands, as opposed to collapsible menus, directly in the results list might benefit users trying to execute multi-step actions.

Finally, our work on re-ranking commands using metrics derived from historical interaction data shows that behavioral signals can be used to improve command rankings (**RG3**). More work is required to merge previous efforts on command recommendation and the use of behavioral data in command retrieval, and through our study we provide a direction for future endeavors in this space.

7 ACKNOWLEDGMENTS

We would like to thank Madeline Kleiner for insightful discussions which helped to shape this research in its early stages, and motivate many of the research questions that were pursued in this paper.

REFERENCES

- [1] Eytan Adar, Mira Dontcheva, and Gierad Laput. 2014. CommandSpace: Modeling the Relationships Between Tasks, Descriptions and Features. In *Proceedings of the Symposium on User Interface Software and Technology (UIST '14)*. ACM, New York, NY, USA, 167–176.
- [2] Eugene Agichtein, Eric Brill, and Susan Dumais. 2006. Improving Web Search Ranking by Incorporating User Behavior Information. In *Proceedings of the Conference on Research and Development in Information Retrieval (SIGIR '06)*. ACM, New York, NY, USA, 19–26.
- [3] Apple Inc. 2017 (accessed September, 2017). *OS X Yosemite: Onscreen help in OS X*. <https://support.apple.com/kb/PH18780>.
- [4] Andrei Broder. 2002. A Taxonomy of Web Search. *SIGIR Forum* 36, 2 (Sept. 2002), 3–10.
- [5] Lydia B. Chilton and Jaime Teevan. 2011. Addressing People’s Information Needs Directly in a Web Search Result Page. In *Proceedings of the Conference on World Wide Web (WWW '11)*. ACM, New York, NY, USA, 27–36.
- [6] Adam Fournay. 2015. *Web Search, Web Tutorials & Software Applications: Characterizing and Supporting the Coordinated Use of Online Resources for Performing Work in Feature-Rich Software*. Ph.D. Dissertation. University of Waterloo, Waterloo, Ontario, Canada. <http://hdl.handle.net/10012/9502>
- [7] Adam Fournay and Susan T. Dumais. 2016. Automatic Identification and Contextual Reformulation of Implicit System-Related Queries. In *Proceedings of the Conference on Research and Development in Information Retrieval (SIGIR '16)*. ACM, New York, NY, USA, 761–764.
- [8] Adam Fournay, Richard Mann, and Michael Terry. 2011. Characterizing the Usability of Interactive Applications Through Query Log Analysis. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 1817–1826.
- [9] C. Ailie Fraser, Mira Dontcheva, Holger Winnemöller, Sheryl Ehrlich, and Scott Klemmer. 2016. DiscoverySpace: Suggesting Actions in Complex Software. In *Proceedings of the Conference on Designing Interactive Systems (DIS '16)*. ACM, New York, NY, USA, 1221–1232.
- [10] G. W. Furnas, T. K. Landauer, L. M. Gomez, and S. T. Dumais. 1987. The Vocabulary Problem in Human-system Communication. *Commun. ACM* 30, 11 (Nov. 1987), 964–971.
- [11] Saul Greenberg. 1993. *The Computer User As Toolsmith: The Use, Reuse, and Organization of Computer-based Tools*. Cambridge University Press, New York, NY, USA.
- [12] Stephen José Hanson, Robert E. Kraut, and James M. Farber. 1984. Interface Design and Multivariate Analysis of UNIX Command Use. *ACM Trans. Inf. Syst.* 2, 1 (Jan. 1984), 42–57.
- [13] Bernard J. Jansen and Danielle Booth. 2010. Classifying Web Queries by Topic and User Intent. In *Extended Abstracts on Human Factors in Computing Systems (CHI EA '10)*. ACM, New York, NY, USA, 4285–4290.
- [14] Bernard J. Jansen, Amanda Spink, and Tefko Saracevic. 2000. Real Life, Real Users, and Real Needs: a Study and Analysis of User Queries on the Web. *Information Processing & Management* 36, 2 (2000), 207 – 227.
- [15] Jiepu Jiang, Ahmed Hassan Awadallah, Rosie Jones, Umut Ozertem, Imed Zitouni, Ranjitha Gurunath Kulkarni, and Omar Zia Khan. 2015. Automatic Online Evaluation of Intelligent Assistants. In *Proceedings of the Conference on World Wide Web (WWW '15)*. ACM, New York, NY, USA, 506–516.
- [16] Julieanne Kost. 2016 (accessed September, 2017). *In-Application Search in Photoshop CC 2017*. <http://blogs.adobe.com/jkost/2016/11/in-application-search-in-photoshop-cc-2017.html>.
- [17] Benjamin Lafreniere, Andrea Bunt, John S. Whissell, Charles L. A. Clarke, and Michael Terry. 2010. Characterizing Large-scale Use of a Direct Manipulation Application in the Wild. In *Proceedings of Graphics Interface (GI '10)*. Canadian Information Processing Society, Toronto, Ont., Canada, 11–18.
- [18] Justin Matejka, Wei Li, Tovi Grossman, and George Fitzmaurice. 2009. CommunityCommands: Command Recommendations for Software Applications. In *Proceedings of the Symposium on User Interface Software and Technology (UIST '09)*. ACM, New York, NY, USA, 193–202.
- [19] Don Norman. 2007. The Next UI Breakthrough: Command Lines. *ACM Interactions* 14, 3 (May 2007), 44–45.
- [20] Daniel E. Rose. 2006. Reconciling Information-Seeking Behavior with Search User Interfaces for the Web. *Journal of the American Society for Information Science and Technology* 57, 6 (2006), 797–799.
- [21] Daniel E. Rose and Danny Levinson. 2004. Understanding User Goals in Web Search. In *Proceedings of the Conference on World Wide Web (WWW '04)*. ACM, New York, NY, USA, 13–19.
- [22] Sofia Stamou and Efthimis N. Efthimiadis. 2010. Interpreting User Inactivity on Search Results. In *Proceedings of the 32nd European Conference on Advances in Information Retrieval (ECIR '10)*. Springer-Verlag, Berlin, Heidelberg, 100–113.
- [23] Jaime Teevan, Eytan Adar, Rosie Jones, and Michael A. S. Potts. 2007. Information Re-retrieval: Repeat Queries in Yahoo’s Logs. In *Proceedings of the Conference on Research and Development in Information Retrieval (SIGIR '07)*. ACM, New York, NY, USA, 151–158.
- [24] Gokhan Tur, Anoop Deoras, and Dilek Hakkani-Tür. 2014. Detecting Out-Of-Domain Utterances Addressed to a Virtual Personal Assistant. In *Proceedings of Interspeech*. ISCA - International Speech Communication Association. <https://www.microsoft.com/en-us/research/publication/detecting-out-of-domain-utterances-addressed-to-a-virtual-personal-assistant/>
- [25] Gui-Rong Xue, Hua-Jun Zeng, Zheng Chen, Yong Yu, Wei-Ying Ma, Wen-Si Xi, and WeiGuo Fan. 2004. Optimizing Web Search Using Web Click-through Data. In *Proceedings of the Conference on Information and Knowledge Management (CIKM '04)*. ACM, New York, NY, USA, 118–126.
- [26] Longqi Yang, Chen Fang, Hailin Jin, Matthew D. Hoffman, and Deborah Estrin. 2017. Personalizing Software and Web Services by Integrating Unstructured Application Usage Traces. In *Proceedings of the Conference on World Wide Web (WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 485–493.