SMS-Based Contextual Web Search

Jay Chen
New York University
jchen@cs.nyu.edu

Brendan Linn
New York University
bl929@nyu.edu

Lakshminarayanan Subramanian
New York University
lakshmi@cs.nyu.edu

ABSTRACT

SMS-based web search is different from traditional web search in that the final response to a search query is limited to a very small number of bytes (typically 1-2 SMS messages, 140 bytes each). SMS-based web search is also a non-interactive search problem where the user has to specify a query and obtain a response in one round of search. Enabling search with with such constraints is challenging. Several search engines have developed SMS-based search capabilities in recent years and many of these search engines are limited in their recognized topics (phone, address, location, weather etc.), involve a human in the loop or apply to only specific types of search queries. We describe a simple generic approach to extracting results for both well-known and arbitrary topics. We have implemented our prototype system SMSFind and demonstrate the effectiveness of our approach. While the underlying mechanisms we present are by no means perfect, we show that our system returns appropriate responses for a range of topics not covered by existing systems.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*Distributed Applications*; H.2.4 [**Database Management**]: Systems—*Query Processing*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Search Process*

General Terms

Algorithms, Design, Experimentation

Keywords

Cell Phones, SMS, Search, Context

1. INTRODUCTION

The rapid growth of mobile devices has made the mobile phone ubiquitous in nearly all parts of the world. The trend is that the personal computing platform is moving away from the traditional desktop and laptops toward more portable and accessible handheld devices. Despite this dramatic shift, the same applications and services that were important to users on desktops, such as web search, remain critical. Web search is facilitated by search engines which index web pages and satisfy queries. Search engines have done extremely well at indexing and returning the appropriate documents for desktop search, and are beginning a push to expand the search ecosystem to handheld devices. This is especially important for developing countries where simple mobile phones are often the first and only computing platform available to the general public. Indeed, mobile phones have already been leveraged for a wide variety of purposes [8, 5, 10, 15, 16] due to their low cost and ubiquity.

The adaptation of search to mobile devices is not completely straightforward. In a desktop search users are able to use a simple search interface to enter queries that often return immediately useful results on the first page. The accuracy of results is generally satisfactory if the desired page is within the top 10 results returned. If the queries entered are vague, users may interactively refine their queries with suggestions from the search engine or iteratively using feedback from the results gathered in each search round. Although users of more powerful smartphones have the luxury of such an iterative search process, the majority of mobile devices are still simple mobile phones that use SMS for search, and the growing global market for SMS search reached 12 million subscribers as of July 2008 [1]. SMS-based search imposes interesting constraints on the problem. First, bandwidth is extremely limited; it is both expensive and tedious for a user to explore even the two most relevant pages returned by a traditional search engine to find the information she wants. Second, the information sought tends to fall in narrower topics (and be smaller in size) than in desktop web search. Returning an entire set of web pages is not acceptable because only a few sentences will fit in the SMS payload. Surprisingly, these terse response are sufficient for the types of search queries SMS search is used for, as typical uses of mobile web search include finding addresses, phone numbers, directions, dates, prices, and short phrases such as the definition of a word. To satisfy these kinds of queries an entire web page of information is not required or even desired.

In this paper, we focus on the problem of "appropriate information extraction": Given a list of search responses to a query by a search engine, how and which 140 bytes should be extracted from the existing search response and associated

MobiHeld'09, August 17, 2009, Barcelona, Spain.

web pages. While existing services for mobile web search (Google SMS [9], Yahoo! oneSearch [21], ChaCha [3]) return appropriate results for a number of topics, including those mentioned above, they are often inconsistent or even meaningless for queries relating to arbitrary topics. For example, Google SMS returns the correct result for the query "Barack Obama birthday", yet it does not return anything for the query "Barack Obama wife". Since both queries have short straightforward answers that are actually present at the top of desktop search results for both Google and Yahoo search engines, this is unsatisfactory. The reason the results are so poor for these topics is because the SMS search services use a limited set of pre-defined topics. These topics are either identified through the use special keywords within the search query such as "directions" or "movies" (Google SMS) or have specialized parsers to determine which of the topics is intended (e.g. querying "AAPL" to Yahoo! oneSearch is a query for information about "stock quote").

The main contribution of this work is to return relevant results without using any pre-defined topics. We built our system SMSFind to extract results for both pre-defined and arbitrary topics of mobile web queries by using an explicit contextual hint entered by the user. Our preliminary results show that although the pre-defined topics in SMSFind may not be as finely tuned as existing systems, for arbitrary context extractors our system returns significantly more relevant results. Thus, our system is complimentary to existing systems that require specialized parsers and data sources. In this paper we begin with an overview of the background and significance of this work and then go on to describe our system architecture and the algorithms, metrics, and heuristics used to extract our results. Finally, we compare the results of example queries obtained from SMSFind with those of Google SMS, Yahoo! oneSearch, and ChaCha.

2. BACKGROUND AND MOTIVATION

Search queries are inherently ambiguous, yet returning a disambiguated result is vital to SMS search queries. In web based information retrieval (IR) disambiguating queries can be done by using context from which the search is being conducted [11, 7]. Loosely, the term context is any additional information associated with a query. The term "context" is highly overloaded even when restricted to the IR space. Context can mean the content of the page currently being viewed by the user [12], the previous activity of the user [19], or even the physical context of the user in the case of ubiquitous and mobile computing [4]. For the purposes of this work we define "context" as the topic that the search result should be about.

Traditional approaches to contextual search may be classified into a few broad categories with their respective definitions of context. One category includes personalization tools which keep track of previous queries or documents viewed or navigation patterns [19, 20]. These types of systems use historical user preferences to refine results to fit the user. While we currently use an explicit context term to bootstrap our system, there is nothing preventing the use of our extraction process with such context construction mechanisms. Another area of research deals with query expansion or reformulation techniques at various stages in the search process to improve recall or precision [2, 14]. These efforts use the context terms somehow as a part of the query itself. While we use the context as a part of the query reformulation to a certain extent we find that it is unnecessary to reformulate or expand heavily to retrieve the desired results. Typically, a "context term vector" is extracted from the context of the search to represent the relative importance of terms in a vector space model [22]. Finally, domain specific search engines use expert knowledge of a domain to improve results [6]. The topic specific extractors in SMS search services are lightweight versions of these expert systems inasmuch as they use some domain specific information to obtain better results.

Contextual SMS search is different from traditional contextual desktop search in several ways. Unlike desktop search, a SMS search does not have a web page from which a reference context may be automatically inferred. To address this shortcoming and still get the benefits of contextual information, we could simply ask the user to explicitly define the context for their query. While this appears to be a simple solution, it is not without pitfalls that must be carefully avoided. First, even in desktop search, users prefer not to spend additional energy formulating queries using advanced search features [17, 18]. This is not unique to mobile search-minimizing the extra cognitive load on the user of a system is a top priority for usability. Second, the small form factor of the device (mobile phone) and the limited bandwidth of the transport medium (SMS) further constrain the possible solutions. Any query and additional information should fit within a single SMS message (less than 140 bytes). Also, the small form factor and low bandwidth make an iterative search process improbable so only a single SMS should be returned. To address these usability issues we adopted the same interface as existing SMS search interfaces (a simple "querybox"), and require the user to append only a single context term as the last word in list of search terms (similar to Google SMS). The challenge is to use this single term hint and a combination of techniques and heuristics to find the most relevant result under 140 bytes and in a single round.

Re-framing this problem in terms of contextual search, requiring only a single word as the context means a vector of length one. With such a short context term vector the traditional algorithms for contextual search do not map directly to the problem. Our problem, however, may be simplified by two observations. First, the results expected by a SMS search are narrower than desktop web search. We can assume that the result should fit in 140 bytes; queries that have no concise and specific answer (e.g. "medieval Japan") are unlikely to occur in our usage scenario. Second, although we are limited to only a single context term, we assume that this context term is useful. Extracting the proper context to use for disambiguation is a research problem in itself, but for us the problem is simplified because the context is explicitly given.

3. SMSFIND OVERVIEW

The architecture of the SMSFind system consists of a server connected to the Internet and to a mobile phone. The client is a user with a mobile phone who sends a SMS message to the number of the server phone which then processes the search query. The search query is dispatched to a general search engine and result pages are downloaded. The server extracts the results from the downloaded pages, and distills them down to 140 bytes. Finally, the server returns the results to the number that issued the request.

The extraction process at the server is the heart of SMS-

Table 1: List of top 10 intermediate N-gram results for the query "the office dwight actor" and their associated metrics prior to filtering heuristics

N-gram	Raw Frequency	Minimum Distance	Mean Rank	Type
wilson	16	1	1.5	name
rainn	16	1	1.25	name
rainn wilson	15	1	1.33	name
dwight schrute	9	2	0.78	name
schrute	9	2	0.77	name
actor rainn wilson	7	0	1.14	mixed
plays dwight	7	2	0.57	mixed
actor rainn	7	0	1.14	mixed
&bull	6	94	4	name
wilson who plays	5	2	0.8	mixed

Find. The input to the system is a search query of the form "<query, context>". where the query represents the actual search terms and the context specifies the type of contextual information that the user expects the system to extract. During the extraction process our system gathers potential results in the form of N-grams from a corpus of webpages, where an N-gram is simply any set of N space delimited terms found within those webpages. The N-grams are measured according to several metrics and then ranked. The most highly ranked result is then returned to the user. In this section we define the relevant metrics used in our algorithms, and detail the process and heuristics used to extract the most relevant results.

3.1 Extraction Metrics

We intend for the algorithms of SMSFind to act as either a front-end to or be incorporated into large-scale search engines. Thus, the metrics used to construct the final rank of each N-gram need to be computationally fast so queries may be answered in real-time. SMSFind pushes as much of the computationally intensive indexing, ranking, and searching down to the underlying search engine as possible. With this in mind, the metrics we have defined for our extraction process are as follows:

Raw frequency - The number of times each N-gram occurs across all result pages. This is similar to the term frequency in the traditional term frequency-inverse document frequency (TF-IDF) weight measure except we are dealing with N-grams and not individual terms. Also, we do not currently normalize the frequency by the document length.

Mean rank - The sum across every occurrence of an N-gram of the rank of the page in which it occurs, divided by the N-gram's raw frequency. We use this measure as the importance of an N-gram similar in spirit to the inverse document frequency component in TF-IDF except our metric incorporates the ranking of the underlying search engine. (In our results some N-grams have a mean rank of less than 1 because the page containing the search engine's results is assigned a rank of zero.)

Minimum distance - The minimum distance between an N-gram and the context term across any occurrences of both. Intuitively, this metric indicates the proximity of the context term defined by the user is to the search query. It is used in disambiguating relevance of two otherwise similarly ranked N-grams.

Type - One of four values: *phrase*, if every 1-gram in the N-gram is an English word (excluding names); *name*, if every 1-gram is a sequence of letters that is not an English word; *number*, if every 1-gram is a sequence of digits; and *mix*, if the N-gram contains 1-grams of different type. We use these types only when we have some notion of the type of result desired for particular topics not for arbitrary topics. These types are not meant to be definitive, but are simply used to show how arbitrary topics in SMSFind may be extended to topics where some domain specific knowledge of the desired result is known.

3.2 Extraction Process

The critical step in SMSFind is the extraction process at the server which distills the search response to 140 bytes. We briefly outline the key steps in our extraction process. The three phases of the extraction process are: *gather*, *distill*, and *coalesce*. We describe each phase and the heuristics involved to develop an intuition about why our extraction is successful.

3.2.1 Gather

The process begins by gathering the corpus of pages from which to extract data. These pages and the search results pages themselves form the working set of SMSFind. Our approach is a simplified version of query rewriting and automated query expansion techniques typically used to improve recall in web search systems. The goal is to gather as many pages as are possibly relevant and to allow the underlying general engine to do the heavy lifting. We build our initial set of pages by downloading the top K search results both with and without the context from a standard search engine (Google). Currently, K is set to 5. While it is possible to use multiple search engines to further increase the recall, for the simple queries we see in SMS search, recall does not appear to be the limiting factor. In our case gathering enough pages to distinguish between N-grams using our extraction metrics is sufficient.

3.2.2 Distill

The distill process of SMSFind is context dependent. SMSFind categorizes contexts into two types: known and unknown. The known context refers to contexts such as phone, weather, address etc. For each known context, we write a context specific information extractor. For example, extracting phone numbers from a web page is fairly straightforward. We outline the different possible ways in which phone numbers may be expressed within a page. For certain types of context such as addresses, writing a context

specific extractor is harder. In this case, we leverage a zipcode extractor and extract the relevant textual information before the zipcode to determine the full address. Search engines have good extractors for known contexts. We focus on describing how distill the relevant information for unknown contexts.

Inputs to SMSFind consist of a list of terms where the last term in the query is the context. For example, in the query "NYU CS department chair", the term "chair" refers to the context of interest. To find the relevant terms for arbitrary contexts we first we perform the distill operation based on the context term in the query. For each parseable (text) page in the corpus the text surrounding the context term is extracted. This is currently done by taking any sentence containing the context term, the sentence before, and the sentence after the context term. We address certain corner cases such as non-sentences using simple heuristics such as delimiting sentences across long whitespaces.

The next step is to *slice* each of these sentences into N-grams of size 1 through 3. Since we are no longer searching for documents of page-sized granularity, it is intuitive for each web page to be further sliced into smaller, more relevant pieces. The maximum N-gram length of 3 is somewhat arbitrary and may be increased, but our system is dependent on the N-gram appearing with some level of frequency for our algorithm to work. Intuitively, longer N-grams will appear less often and may be below the threshold for which we distill N-grams. This can be mitigated if we merge the counts for synonymous N-grams, but we leave that for future work. Our next step is to count the raw frequency of the N-grams in our corpus, and measure the distance from the context term at which each N-gram occurrs and record the minimum. An example of the type of data we have at this point is shown in Table 1. In this example query "the office dwight actor" should return the response "rainn wilson" as highlighted in the table. Note that this type of example is exactly in the range of queries that we are interested in, it is too rare for a custom extractor and common enough to be detectable by our system. From the list of N-grams we can observe that after slicing most of the top results are highly relevant to the query according to our metrics. However, simply naively re-ranking according to any *single* metric we have defined will not return the correct result. At this point the slicing and measurement process for N-grams this measurement process is complete, and SMSFind begins distilling the results using several heuristics and tunable cutoffs.

Heuristics - After the potential N-grams are gathered, we employ several heuristics to remove unlikely candidates. The first heuristic we use is to remove N-grams with low frequency (< 4). These are unlikely to be relevant since they do not occur commonly enough. The next heuristic we use is to remove N-grams with high minimum distance (> 10). The intuition here is that while these N-grams appear often enough, they never appear close enough to the context term to be relevant. We also remove N-grams with minimum distance 0 to avoid the echoing the context term back to the user. Currently, the thresholds for these heuristics are hand tuned; the generalization of these parameters is a topic we leave for further study. Next, we perform additional heuristics for distilling "English phrases" using a simple 1MB dictionary of terms and the "Web 1T 5-gram Version 1" dataset from the Linguistic Data Consortium (LDC) [13] which provides the web shingle frequency information. Again, the cutoffs are tunable parameters in our system and are set at values that we observed to work well. We also remove 1-grams which are extremely short (< 3 letters) and N-grams consisting only of pronouns or hypertext tags to reduce noise. N-grams which are subsumed by longer N-grams are also removed. We do not currently incorporate the metrics for the subsumed N-grams into the larger N-gram because it is still unclear what rank-aggregation algorithm would be the most beneficial. Finally, we remove the N-grams which overlap with the original query (> 1 term in common) to avoid echoing back the query.

3.2.3 Coalesce

In this final phase the N-grams are first categorized by their type. The type is used only for contexts that we have some concept of to filter out irrelevant types. E.g. for a context of phone, we would filter out N-grams of non-numeric type. For contexts that we have no expert knowledge about, the type is simply ignored. The ranking process is performed by first bucketing the N-grams by their minimum distance, and ranking from lowest minimum distance first. Then for each bucket, the N-grams are ranked by their mean rank. Continuing from our earlier example, the final result after applying all of the heuristics is "rainn wilson". In the examples section we take a look at more examples comparing SMSFind with existing SMS search services and begin exploring the limitations of our approach.

4. IMPLEMENTATION

SMSFind is implemented in only 300 lines of Python code and uses publicly available parsing libraries. We have not yet conducted detailed benchmarks on our implementation or performed any optimizations. On a 1.8Ghz Duo Core Intel PC with 2 GB of RAM and a 100Mbps connection SMSFind generally returns results within 5-10 seconds. This is inclusive of the time required to fetch query results from Google and download web pages referred to by the results.

5. PRELIMINARY EVALUATION

First, to give a comparison of the difference between the results found by SMSFind and the existing systems we show some example queries and their results returned. In Table 2 we compare SMSFind to similar services, including Google SMS, Yahoo! oneSearch, and ChaCha.[1] For the first example we show how well the different systems work for a known context to get a baseline to compare against for SMSFind. For the query "abraham lincoln birthday", all three automated services were successful to some extent. In the case of GoogleSMS and Yahoo! oneSearch the custom extractors recognized the "birthday" keyword, and were able to return correct results. Interestingly, we did not have a topic-specific extractor implemented for "birthday" in SMSFind, yet our results are correct. ChaCha also returned excellent results for this first query.

[1]Note that ChaCha is a human-mediated service. It refuses all of the queries as they appear in Table 2. The output under the ChaCha column was received only after rephrasing the queries in natural language, e.g. "when is Abraham Lincoln's birthday?" Requiring users to rephrase queries in natural language represents in some cases an additional constraint upon the user if they do not wish to use in this extended format.

Table 2: Comparison of results for example queries between search services

query	SMSFind	Google SMS	Yahoo! oneSearch	ChaCha
abraham lincoln birthday	february 12	Abraham Lincoln Date of Birth: 12 February 1809	Celebrate! Holidays In The USA - Abraham Lincoln's Birthday (February 12)	Abraham Lincoln was born on February 12, 1809, in Hodgenville, Hardin County, Kentucky.
google ceo	eric schmidt	(no results)	(stock lookup for ticker "CEO")	Eric Schmidt is the Chairman of the Board and Chief Executive Officer of Google
lakshmi subramanian phone	212 998 3485	(no results)	News: Oops! 2 Suspects Hail Wrong Car	They are located at 6189 N Pinnackle Ridge Dr. Tucson, AZ 85718-3502

Next, we show how SMSFind works with an arbitrary topic example that is simple, but does not have terms that are common enough for a custom extractor to be built. The second query is: "google ceo". This example is where the arbitrary context extractor of SMSFind distinguishes itself. By using ceo as the context SMSFind was able to return a highly relevant result. Ironically, Google SMS returned no results, probably because it has no custom extractors for the "ceo" keyword. Yahoo! oneSearch returned a false positive result, determining that the query was tied to the stock quote topic and interpreted "ceo" as a stock symbol. The human expert at ChaCha once again had no trouble returning the correct result. For the third query "lakshmi subramanian phone", SMSFind recognized the context term as one expecting a numerical answer, and filtered out textual results properly to return the correct result. Google SMS again returned no results, even though the first web page returned in a desktop Google search contains the answer. Again, Yahoo! oneSearch's top result was bizarre and unhelpful. Even the human-mediated ChaCha appears to have misinterpreted the query to be about a private residence.

Finally, we ran SMSFind on queries adapted from Google Trends queries to evaluate performance. Google Trends is an hourly enumeration of 100 queries deemed by Google to be popular (although the company does not disclose how it ranks them). The queries typically reflect the day's news and popular culture, so they are broadly relevant for evaluating SMSFind. However, a majority of Google Trends queries are open-ended in a way that SMS queries presumably would not be. We modified the Google Trends queries from May 5, 2009 by adding a focusing keyword to obtain a query useful to evaluating an SMS search tool. For example, a manual web search revealed that the query "kryptos" was popular on May 5 likely due to a recent press coverage of a piece of artwork. Accordingly, the query was modified to "kryptos artist" before sending it to SMSFind. We modified 31 of the 100 queries in this manner. The remaining 69 queries were too vague for us to modify in a meaningful way.

Overall, SMSFind returned a correct response for 22 of the 31 queries ($\approx 71\%$). and when it did return a correct response, the mean rank of the response was ≈ 3.36. Table 3 shows the best result returned by SMSFind with accompanying rank for the 22 queries where SMSFind returned a correct response. SMSFind answers "person" and "date" queries well, although not perfectly (note the low rank of the correct result for "nba mvp", for example). Broader queries, such as asking for a symptom of a disease, perform less well.

Table 4 shows the queries that SMSFind failed to answer. Although our test corpus is fairly small we can observe that certain queries and contexts do not perform as as others presumably due to their low frequency count.

6. CONCLUSION & FUTURE WORK

SMSFind shows that very simple algorithms that leverage existing search engines are capable of returning results that perform better for a wide variety of topics that custom extractors are not suited for. While human-in-the-loop and NLP techniques may provide equal or better precision, they come at the cost of computation or human intervention and do not scale as well as this simple approach. We have shown that our system performs well using only a single additional context hint in the query. A more detailed evaluation is required to determine the precise algorithms and operating parameters that work best, but the overall approach is both novel and promising.

Our approach currently implements no additional optimizations. Some very simple optimizations and improvements that could lead to even better performance include noise reduction using various filtering techniques, and categorization for specialized parsers and databases such as those used by the existing SMS search systems. Another exciting and unexplored avenue is in adapting our approach to international markets that use other languages or scripts. Finally, we currently only return the top N-grams as our result; it would be simple to return surrounding sentences or multiple results to the user. Regardless of the future direction, we expect that as mobile phones become more powerful the hardware capabilities may increase, but the need for short focused bits of information will remain.

7. REFERENCES

[1] Critical Mass: The Worldwide State of the Mobile Web. *The Nielsen Company*, 2008.

[2] P. Bruza, R. McArthur, and S. Dennis. Interactive Internet search: keyword, directory and query reformulation mechanisms compared. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 280–287. ACM New York, NY, USA, 2000.

Table 3: SMSFind successful Google Trends queries

Original Query	Modified Query	Correct Result (if returned)	Mean Rank of Correct Result
kryptos	kryptos artist	artist james sanborn	1
james sanborn	james sanborn sculpture	kryptos sculpture	4
shia labeouf mother	shia labeouf mother	shayna saide	4
renardo sidney	renardo sidney school	ucla	8
celiac disease	celiac disease symptom	constipation	6
celiac disease	celiac disease symptom	malabsorption of nutrients	7
maggie gyllenhaal	maggie gyllenhaal husband	peter sarsgaard is the husband	2
mexican flag	mexican flag colors	green white and red	1
trent reznor	trent reznor girlfriend	mariqueen maandig	2
swine flue	swine flu school	students back at nyc	2
filene s basement	filenes basement location	boston	6
selena roberts	selena roberts book	arod	4
selena roberts	selena roberts book	alex rodriguez	7
mexican desserts	mexican dessert	ancho	5
battle of puebla	battle of puebla date	1862	1
nba mvp	nba mvp	lebron james	4
danny gans	danny gans death cause	unknown	2
jon and kate plus 8 drama	jon kate plus 8 star	jon gosselin	2
rihanna	rihanna boyfriend	boyfriend chris brown	1
john mayer	john mayer girlfriend	girlfriend scheana marie	2
bob schimmel	bob schimmel wife	melissa	2
rolex submariner	rolex submariner price	16610	1

Table 4: SMSFind failed Google Trends queries

Original Query	Modified Query
peter sarsgaard	peter sarsgaard wife
bruce pearl	bruce pearl fiancee
blackberry curve	blackberry curve cost
star wars day	star wars day
mega millions	mega millions winner
nj marathon	nj marathon winner
jack kemp	jack kemp funeral
timothy wright	timothy wright funeral
steve-o demise and rise	steve-o demise and rise network

[3] ChaCha. http://www.chacha.com.

[4] G. Chen and D. Kotz. A survey of context-aware mobile computing research. Technical report, Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College, 2000.

[5] CommCare. http://www.dimagi.com/content/commcare.html.

[6] W. Dorda, T. Wrba, G. Duftschmid, P. Sachs, W. Gall, C. Rehnelt, G. Boldt, and W. Premauer. ArchiMed: a medical information and retrieval system. *Phlebologie*, 38(1):16–24, 2008.

[7] L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, and E. Ruppin. Placing search in context: The concept revisited. *ACM Transactions on Information Systems*, 20(1):116–131, 2002.

[8] FrontlineSMS. http://www.frontlinesms.com/.

[9] Google SMS. http://www.google.com/sms.

[10] InSTEDD GeoChat. http://instedd.org/geochat/.

[11] R. Kraft, C. Chang, F. Maghoul, and R. Kumar. Searching with context. In *Proceedings of the 15th international conference on World Wide Web*, pages 477–486. ACM New York, NY, USA, 2006.

[12] S. Lawrence. Context in web search. *IEEE Data Engineering Bulletin*, 23(3):25–32, 2000.

[13] Linguistic Data Consortium. http://www.ldc.upenn.edu.

[14] M. Mitra, A. Singhal, and C. Buckley. Improving automatic query expansion. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 206–214. ACM New York, NY, USA, 1998.

[15] OpenRosa. http://www.openrosa.org/.

[16] M. Paik et al. The Case for SmartTrack. *IEEE/ACM Conference on Information and Communication Technologies and Development (ICTD)*, 2009.

[17] G. Pass, A. Chowdhury, and C. Torgeson. A Picture of Search. In *First Intl. Conf. on Scalable Information Systems*, 2006.

[18] D. E. Rose and D. Levinson. Understanding User Goals in Web Search. In *WWW*, May 2004.

[19] X. Shen, B. Tan, and C. Zhai. Context-sensitive information retrieval using implicit feedback. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 43–50. ACM New York, NY, USA, 2005.

[20] J. Teevan, S. Dumais, and E. Horvitz. Personalizing search via automated analysis of interests and activities. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 449–456. ACM New York, NY, USA, 2005.

[21] Yahoo One Search. http://mobile.yahoo.com/onesearch.

[22] C. Yu, K. Lam, and G. Salton. Term weighting in information retrieval using the term precision model. *Journal of the ACM (JACM)*, 29(1):152–170, 1982.