SMSSearch: An Enhanced Mobile Searching Application

Manoj Kapil[1], R.P.Yadav[2]

1.Research Scholar, Magadh University Bodh Gaya Bihar

2.Professor & Head (Retired) Magadh University Bodhgaya

ABSTRACT

SMS-supported mobile searching applications are gradually more widespread around the globe, particularly in current scenario among users with low-end mobile devices. Web exploration is one of the main and valuable of these applications. SMS-supported web exploration takes unstructured queries and returns web snippets via SMS. This application permits users with low-priced mobile phones and no data plan to hit into the enormous amount of information on the web. SMS-supported web exploration is a demanding problem since the channel is both tremendously constricted and costly. Preferably, together the query and the response fit within an SMS. This chapter presents the blueprint and execution of SMSSearch, an SMS-supported search application that facilitates users to achieve exceptionally crisp (one SMS message) search responses for queries across random keywords in one round of communication. SMSSearch is intended to balance existing SMS-supported search applications that are moreover inadequate in the keywords they identify or engage an individual in the loop.

Specified an *unstructured* search query, SMSSearch, uses a usual search engine as a back-end to extract a number of search responses and uses a mixture of information retrieval procedures to take out the most suitable 140-byte snippet as the final SMS search reply. We illustrate that SMSSearch returns suitable responses for 62.5% of Mahalo search queries in our experiment set; this precision rate is high given that Mahalo make use of a human to respond the identical queries. We have also installed a pilot

edition of SMSSearch for make use of with a little featured group and a bigger scale open beta in Meerut to investigate our application and contribute to our understanding.

Keywords: SMSSearch, Mobile Search Application, Web Exploration, SMS, Unstructured Query, Mahalo

1. Enthusiasm

The outstanding development of the mobile phone market has provoked the blueprint of new types of mobile information applications. With the development of Twitter [14], SMS GupShup [11] and other community messaging networks, the precedent few years have observed a rising occurrence of Short-Messaging Service (SMS) based applications and services. SMS-supported applications are also more and more frequent in poor nations. Regardless of the growing influence of mobile devices with the introduction of "smart phones", a significant portion of mobile devices in current scenario are still straightforward economical devices with restricted processing and communication potential. Due to a combination of social and economic issues, voice and SMS will expected prolong to stay the main communication channels existing for a non-trivial portion of the people in current scenario. For any SMS-supported web service, efficient *SMS-supported search* is an important building block. SMS-supported search is a speedily rising worldwide market with over 12 million subscribers as of July 2008 [24]. An SMS message is limited to 140 bytes, which severely confines the amount of information in a search reply. SMS-supported search is also non-interactive due to the search reply time; anecdotally [10], existing SMS-supported search engines acquire on the order of tens of seconds [6, 16] to several minutes per response [1]. Still not including the 140-byte SMS size restriction, extendedoring conventional web search to mobile devices is a difficult problem due to the small form issue and highly intermittent environment. Unlike desktop search, clients on mobile devices hardly ever have the comfort of iteratively refining search queries or filteri through pages of outcomes for the information they want [37]. In this chapter, we concentrate on the difficulty of *SMS-supported search*: *how does a*

mobile client efficiently explore the web using one round of communication where the search reply is constrained to one SMS message?

Despite the fact that we do not know the internals of existing SMS search algorithms, we can examine from the client interface and credentials that existing automated applications for SMS web search such as Google SMS [6] and Yahoo! oneSearch [16]) support clients to enter queries for an amount of pre-defined keywords. These pre-defined keywords are either identified during the utilization of particular keywords within the search query such as "descrbe" or "planet" (e.g. Yahoo SMS: "describe planet") or have specific parsers to find out which of the keywords is intentional (e.g. querying "RAW" to Yahoo! oneSearch is a query for information about "intelligence unit"). Mahalo [1], a latest SMS-supported search engine, employs peoples to explore the web and respond queries in an SMS reply. TradeNet, now called eSoko [3], is a mobile marketplace platform in Ghana that would need an SMS based search service as well. Immediate access to small bits of information is the enthusiasm at the back all of the existing SMS based search applications and of this effort. None of the existing automated SMS search applications is an absolute resolution for search queries across random keywords. Similar to conventional web search queries, SMS search queries undergo the *long extended trend*: there exists an extended extended of search queries whose keywords are not accepted (e.g. "what are schooling gift ideas?" or "what chemicals are in mosquitoes repellent"). We confirm that this certainly is the case in our sample of Mahalo questions where only 27% of the queries in our data set are verticals (as defined by Google SMS) and 73% are long extended. In this chapter, we explain the blueprint and execution of SMSSearch, an SMS-supported search engine specifically intended for the long extended of search queries that are stretched across a large collection of keywords. These keywords correspond to the queries in a mobile search workload that are not responded by existing domain specific verticals. SMSSearch is intended to incorporate into an existing SMS search application to respond queries for unsupported long extended keywords. Given a query, SMSSearch employs a conventional search engine

as a back-end to extract a number of search outcomes and take out the suitable 140 bytes as the SMS search reply. Section 5.2 further elucidates how vertical and long extended queries are defined in this effort.

2. SMSSearch Setback

SMSSearch uses a grouping of well-known information retrieval procedures to deal with the suitable information mining problem. SMSSearch is intended for *unstructured queries* supporting a related design as a customary search engine query. The key proposal of SMSSearch is that significant SMS queries normally include a term or a group of successive terms in a query that supplies a *clue* as to what the client is searching for. The clue for a query can either be unambiguously supplied by the client or automatically derived from the query. SMSSearch uses this clue to deal with the information mining trouble as follows: Known the top N search replies to a query from a search engine, SMSSearch mines *snippets* of content from within the neighborhood of the clue in each reply page. SMSSearch scores snippets and ranks them across a multiplicity of metrics. The design of our SMS search application (Figure 1) consists of a query server that handles the authentic search query and outcomes, and an SMS gateway that is accountable for communication between the mobile clients and the query server. The client is a user with a mobile phone who launch an SMS message to the short code (a particular number, shorter than full telephone numbers used to deal with SMS and MMS messages) for our application, which reaches at the SMS gateway and is then send out to our server for processing. At our query server the query is then sent to a common search engine and answer pages are downloaded. The query server mines the outcomes from the downloaded pages, and returns them to the SMS gateway that sends it back to the client that issued the call.

Figure 1: System architecture.

2.1 Identified Verticals vs Long Extended

Certain SMS based queries are most excellent responded by querying known verticals (e.g., train, directions, agriculture). Google SMS search may be observed as a covering around its search application for a fixed collection of known groups such as mobile numbers, agriculture, train information, address etc. Our whole SMS search application consists of suitable parsers and vertical redirectors for a small number of known groups (mobile numbers, agriculture, addresses). For example, *trainenquiry.com* and *sulekha.com* are examples of such verticals for weather and yellow pages. For these groups with existing verticals, producing an SMS search reply needs a straightforward alteration of the query into a suitable database query (or filling a form) at the related vertical web portal.

The center of SMSSearch is to hold long extended queries that do not fit into verticals. Managing queries for verticals is a comparatively easy if monotonous process and suffers from speedily retreating returns. Our entire SMS search application supports a fundamental set of vertical keywords that is a subset of the Google SMS keywords. As an element of this structure, the SMSSearch algorithm is positioned at the back of a categorizer that first resolves whether a specified query belongs to an employed vertical based on the existence of defined keywords or if it should be sent to SMSSearch.

2.2 SMSSearch Exploration Problem

SMSSearch is intended for unstructured search queries across random keywords. Given any query, SMSSearch first employs an existing search engine as a back-end to acquire the top few search outcome pages. Using these pages the remaining problem is to parse the textual content to acquire the suitable search answers that can be reduced to one SMS message. This is basically a problem of dewordining suitable reduced snippets of text across the outcome pages that form candidate SMS search answers. We define a *snippet* as any uninterrupted stream of text that fits inside an SMS message. SMSSearch uses a clue for every query as a significant clue to mine every outcome page for suitable text snippets. A *clue* refers to either a word or a collection of consecutive words within the query that is approximately analytic of what type of information the client is searching for. Specified that the clue will emerge in the answer page, SMSSearch uses the neighborhood of the text around the clue to mine for suitable textual snippets. To elucidate our problem and algorithm we presume that the clue is given by the client unambiguously, but afterward we talk about how the clue may be automatically mined from usual language questions. The SMSSearch exploration problem can be described as follows:

Given an unstructured SMS search query in the form of <query, clue> and the textual content of the top N search answer pages as returned by a search engine for “query”, mine a reduced set of text snippets from the answer pages that fit within an SMS message (140 bytes) that give a suitable search reply to

the query. Where the clue is a word or collection of words found within the query. This problem definition unambiguously presume that the clue is specified for every query. Existing SMS-supported search applications like Google SMS have a similar unambiguous prerequisite where a keyword is specified as the very last word in a query; the differentiation being that the existing applications only maintain a fixed set of keywords whereas SMSSearch allows random clues.

3. SMSSearch Exploration Algorithm

In this part, we depict our algorithm to mine snippets for any given unstructured query of the form <query, clue>.

3.1 Fundamental Design

Search queries are essentially vague and a universal technique to disambiguate queries is to use supplementary contextual information, from which, the search is being performed [26, 31]. Insecurely, the word "context" is any supplementary information linked with a query that gives a helpful clue in providing a targeted search answer for a query [21,32]. Similar to these mechanisms, SMSSearch uses an unambiguous clue so the snippet mining algorithm can classify the estimated position of the preferred information in a search answer page. We encourage the use of a clue using a straightforward instance of a long extended query. Consider the query "Barack Obama wife" where the word "wife" correspond to the clue. When we give this query to a search engine, most search outcome pages will contain "Sonia" or "Sonia Gandhi" or "Sonia Sonia Sharma" or "Sonia Rakesh Sharma" within the neighborhood of the word "wife" in the text of the page. For this query, to conclude any of these as suitable search answers, SMSSearch will explore the neighborhood of the word "wife" in every outcome page and look or frequently occurring *n-grams* (where n corresponds to one to five repeated words). For example, "Sonia Gandhi" is a n-gram that is a 2−gram. A straightforward algorithm to conclude the acceptable answer to

this example query is to yield all accepted n-grams within the neighborhood of the clue and grade them based on different metrics (frequency, distance etc.). On the other hand, outputting frequently occurring n-grams as search answers is only suitable when the authentic search answer for a query is a 1 − 5 word answer. For a number of general SMS-search queries, the authentic suitable search answer is embedded in a sentence or a collection of few sentences. In such cases, we require to mine entire snippets of text as a search answer as contrasting to just n-grams. SMSSearch makes a unambiguous distinction between *n-grams* and *snippets*. Although both correspond to uninterrupted sequences of words in a document, a n-gram is particularly short in length (1 – 5 words), while a text snippet is a sequence of words that can fit in a single SMS message. In our SMSSearch algorithm, n-grams are employed as an middle unit for our statistical techniques while snippets are employed for the final grading since they are suitably sized for SMS. We subsequently depict the fundamental SMSSearch algorithm. Consider a search query (Q, C) where Q is the search query including the clue word(s) C . Let O_1, . . . O_N correspond to the textual content of the top N search answer pages to Q. Given (Q, C) and O_1 . . . O_N , the SMSSearch snippet mining algorithm consists of three steps:

Neighborhood Mining: For each outcome page Oi, SMSSearch explores for every appearance of the clue word C and mines a textual neighborhood around the clue that corresponds to a candidate snippet of length covering one SMS message in either side of the clue. For each snippet, we mine all exclusive n-grams with up to 5 words. The selection of the limit 5 is provoked by the reality that the Linguistic Data Consortium (LDC) [9] publishes the web occurrence of each n-gram with up to 5 words.

N-gram Grading*:* We grade the n-grams based on three metrics: distance to the clue, frequency of occurrence, and average grade of the outcome page. We also use the comparative infrequency of a n-gram on the web to standardize the n-gram grading.

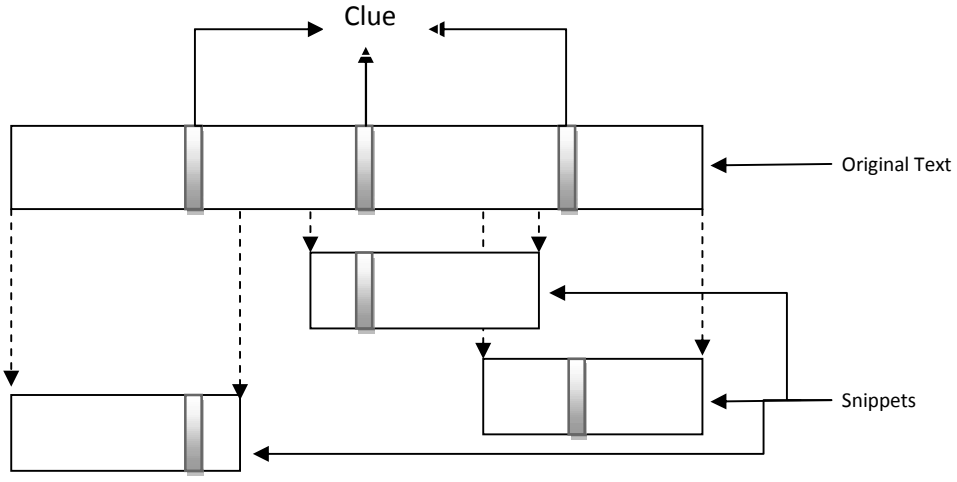
Snippet Grading: We define the grade of any snippet as a collective summation of the top-few graded n-

grams within the snippet. Among all snippets, we conclude the top-graded snippet(s) as the search answer. We now intricate upon these three steps.

3.2 Neighborhood Mining

Given a query (Q, C) and its outcome pages O_1, . . . O_N , SMSSearch first mine snippets around the clue C in each of the pages. For each page Oi, we find every happening of the clue C in the page. Each snippet is up to 140 bytes long and the clue is as centered as much as the neighboring text allows. We establish that delineating snippets by sentence boundaries direct to many corner cases due to sound that could distort the statistical outcomes. The snippets are then combined if they overlap to keep away from double counting into *snippet tiles* (Figure 2). These snippet tiles structure the foundation of all additional measurements and calculations, and it is only within these snippet tiles that the final outcome is mined. From a realistic perspective of not needing to download a number of pages and having sufficient variety to mine n-grams and snippets, we establish that a value of N = 10 works fit. We mine the text from these web pages by filtering out all characters, hypertext tags, and non-ASCII ciphers. The outcome is simple text that is similar

to what would be provided by an ordinary browser.



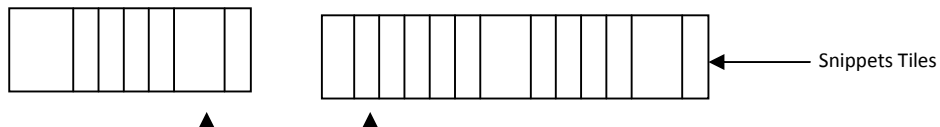


Figure 2: Snippet creation, aggregation into snippet tiles, and n-grams.

3.3 N-gram Grading

N-gram grading is a significant step in the SMSSearch snippet mining algorithm. Given any snippet mined around the clue, the first step in our n-gram grading algorithm is to assemble all probable n-grams in the snippet. Table 1 demonstrates briefly how the n-grams are produced. The objective of the n-gram grading algorithm is finding the n-grams that are most probable to be associated to the accurate answer. The underlying principle of our n-gram grading algorithm is that any n-gram that satisfies the subsequent three properties is potentially associated to the suitable answer for a query with a specified clue:

1. the n-gram become visible very often around the clue.

2. the n-gram become visible very close to the clue.

3. the n-gram is not a universally used popular word or expression.

As an example, the n-gram “Sonia Gandhi” is not a universally used expression and come into sight

Table 1: Slicing example for the text “the quick fox jumped over the lazy dog”, clue = “over”.

N-Gram	Occurrence	Least distance
“the”	2	1
“the quick”	1	3
“the quick fox”	1	2
“quick fox jumped”	1	1
-----	-	-

comparatively often and in close proximity of the clue "wife" for the top search answer pages to the query "Rajiv Gandhi wife". Consequently, this n-gram is extremely significant for the search answer for the query. For each exclusive n-gram in any snippet, we calculate three autonomous measures:

Occurrence - The number of times the n-gram arises across all snippets.

Average grade - The summation across every happening of a n-gram of the PageGrade of the page where it occurs, divided by the n-gram's raw occurrence. This is to include the grading scheme of the fundamental search engine in our overall grading function. (Some n-grams have a raw average grade of less than 1 because the page enclosing the search outcomes is alloted a grade of 0.)

Least distance - The least distance between a n-gram and the clue across any happenings of both. Spontaneously, this metric shows the proximity of the clue defined by the client is to the search query. It is used as an element of our overall grading function to let the client clue to disambiguate two otherwise likewise graded n-grams.

An instance of the metrics we have at this point is shown in Table 2. In this instance, the query "wearable computer for head-mounted display" should return the answer "Google Glass" as highlighted in the table. Note that this illustration is precisely in the range of queries that we are concerned in, it is too exceptional for a conventional miner and universal enough to be demonstrable by our scheme. From the catalog of n-grams we can scrutinize that after slicing, most of the top outcomes are highly significant to the query according to our metrics.

Table 2: list of top 5 n-grams results for the query "wearable computer for head-mounted display" and their associated raw metrics prior to normalization.

N-Gram	Occurrence	Least distance	Average Grade
Google	12	1	1.2
Glass	12	1	1.14
Google Glass	11	1	0.89
Wearable computer	10	2	1.08
Head-mounted	9	2	1.76

Filtering n-grams: Before grading n-grams, we filter the set of n-grams based on the three measures: occurrence, average grade and least distance. A n-gram should have a least occurrence and should be within a definite least distance of the clue to be measured. For N = 5, we set a least frequency bound of 3 and a least distance threshold of 10; we prefer these thresholds experimentally based on manual investigation of n-grams across sample queries. Correspondingly, we pay no attention to all n-grams with a very low average PageGrade.

Grading n-grams: Correlating comparative significance to any of the metrics or idealistically grading based on a particular metric is not suitable. To combine the different metrics into a single metric we carry out two straightforward steps. First, we standardize the raw occurrence, average grade, and least distance to a homogeneous distribution within the range between 0 and 1. We indicate the three standardized scores of a n-gram *s* as occr(s), avggrade(s), leastdist(s). Second, the overall grading score of a n-gram *s* is a linear combination of the three standardized grades:

$$\mathbf{grade(s) = accr(s) + avggrade(s) + leastdist(s)} \qquad \mathbf{(1)}$$

We use the grading score to grade all n-grams related with a query. We need to regard as one significant feature in the standardization of the occurrence score. If two n-grams, *t* have the same

occurrence measure but if n-gram *s* has a much lesser web occurrence than n-gram *t* (*s* is unusual than *t*), then *s* desires to be higher graded than *t*. We use the "Web 1T 5-gram Version 1" dataset from the LDC to attain the occurrence for any n-gram and calculate its standardized occurrence.

3.4 Snippet Grading Algorithm

If the reply to a query is a very small answer of a few words, then the finest SMS based search answer is to output all the top-graded n-grams related with a query. Though, given a query, it is hard to conclude whether the reply is embedded in a single n-gram or is truly a grouping of multiple n-grams. In this scenario we yield the most excellent snippet under the expectation that the reply is embedded in the snippet and the client can understand it. The n-gram grading algorithm cannot be extended to grading snippets since approximately all of the snippets are exclusive, and their occurrence measure will be 1. We enlarge the n-gram grading algorithm to compute the grade of snippets based on the n-grams present within a snippet. In addition, different snippets may enclose different number of n-grams that may bring in a prejudice in the grading function. To keep away from such a prejudice, we bring in a top-R n-grams based grading function.

Snippet grade: Consider a snippet S with a set of n-grams O = $\{o_1, \ldots o_m\}$ with corresponding n-gram grades grade(o_1), . . . grade(o_m). Let o_{i1} , . . . o_{iR} correspond to the top R graded n-grams in O . Then the grade of snippet S is:

In other words, we define the grade of a snippet based on the collective grade of the top-R graded n-

grams within the snippet. In practice, we select R = 5. In the *snippet grading* stage, we conclude the maximum graded snippet is the most significant answer to the original query. Remember that our snippets were intended to be under 140 bytes, but the snippet tiles may in fact be longer depending on overlaps during the integration process. To find the most excellent snippet, we first split each snippet tile into snippets using a 140 byte sliding window across every snippet tile that compliments word restrictions. We then score each snippet based on the summation of the top R n-grams and return the top scoring snippet as the final outcome. In the estimation, we show that grading n-grams first before grading snippets is important for enhanced precision; in other words, straightforwardly scoring snippets from the web page outcomes without using n-grams performs very disappointingly in practice.

3.5 Clue Mining from the Query

We have thus far presumed that every query is related with a clue, and for unstructured queries it is a usual tendency for the client to enter the clue either at the beginning or at the end. Though, even if questions are entered in usual language layout, mining the clue automatically is not difficult. We illustrate a straightforward rule-based approach for originating the clue for usual language question-style queries as a verification of model. The approach we use is alike to surface pattern matching techniques [38]. A quick investigation of a sample of 10,000 queries from Mahalo SMS-supported search queries exposes that a huge portion of queries use universal syntactic structures where the clue is straightforwardly identified. Nearly 96% of Mahalo search queries are English queries with 15 words or fewer (80% of queries include fewer than 10 words). Based on the constitution of the question, we have classified a universal group of questions and their corresponding alteration rules to conclude the clue. We found that that almost 55% of the queries started with the word “what”, of which, over 85% of the queries are in standard forms (for e.g. “what is”, “what was”, “what are”, “what do”, “what

does"). For each one of these blueprints, we can write a straightforward alteration rule to mine the clue from the corresponding sentence that is typically either instantaneously after the question or toward the end of the sentence. For example, for the query "what is a structured programming language by Dennis Ritchie", the "what is X" pattern employs a simple matching rule to mine the clue word "keyboard" (if "a" is disregarded as a stop word). The remaining words minus stop words are then used as the query, and the final <query, clue> is: <"Dennis Ritchie", "structured programming language">.

4. Execution

The foundation SMSSearch algorithm is executed in only 597 lines of ASP code and uses widely accessible parsing libraries. We have not executed any optimizations or caching, but SMSSearch normally returns outcomes within 4-10 seconds while running on a 2.4Ghz Intel Core i3 PC with 4 GB of RAM and a 2 Mbps broadband connection. This response time is dominated by the time essential to obtain query outcome from Google and download web pages referred to by the outcomes. To install SMSSearch as a search application we employed a front-end to send and receive SMS messages. We setup a SMS short code, and direct all SMS requests and replies to and from our server machine running the application across a Samba 75 GSM modem and Kannel an open source SMS Gateway [8]. As a verification of idea, and to advance the overall usability of our organization we have also employed interfaces to several basic verticals as a part of the application including: agriculture, describe, local business results, and news. Each of these interfaces to verticals is under 200 lines of ASP code.

5. Assessment

Entirely assessing such an application is non-trivial; a huge set of reasonable queries must be asked,

and replies must be evaluated either by evaluators or the clients themselves. We obtain a practical set of queries by transforming actual Mahalo queries. We then assess our structure performance in complete words to explore the restrictions of our structure and probable enhancements. In our assessment, the replies returned by SMSSearch are evaluated acceptable *if the all of the exact answer words emerge anywhere in the returned snippet*. The accurate answer were first determined by three evaluators who came up with exact answers by manual scrutiny of each question, referring to the Mahalo reply for reference. Occasionally even the Mahalo reply was considered wrong by the evaluators and a different reply was concluded online and used in its place. There were also queries that had a number of suitable answers, we consider all of those "accurate" in our assessment. This is the straightforward rating structure we use throughout the assessment. We do not at present grade our scores based on other significant features such as readability or precision. For a full-fledged question/reply structure, including methods to advance readability [30] would likely be essential.

5.1 Statistics

Our set of queries consists of a mass of 10,000 actual SMS search queries scraped from Mahalo's [1] unrestricted website on August 22, 2012. These queries are in Normal Language query layout. In comparison to the logs evaluated earlier studies we establish in our statistics an average of 8.90 words per query and 43.17 characters per query, in contrast to 3.05 and 18.48 respectively reported in [39]. The TREC [13] query/reply track datasets are based on search engine queries, and TREC-9 datasets were authentic clients' queries. Conversely, it is ambiguous whether they were collected from mobile devices. From earlier studies it is obvious that mobile search logs have significant differentiations across lots of characteristics depending on the search means and device [28, 29]. Queries from Mahalo are mobile (either voice or SMS) and the replies are created by a peoples and returned via SMS. Consequently, we selected to use the Mahalo dataset due to its high practicality for query types and

their allocation across keywords in a mobile background. We merely use all 10,000 queries for our comprehensive investigation of query length and keyword distributions. Since our assessment consists of both rewriting, and manual evaluation of queries and replies, both of which are effort intensive, we were not capable to carry out comprehensive assessment with the full set of 10,000 queries. In its place, an arbitrarily chosen 2,000 query subset is used for our more comprehensive manual assessments. Out of these 2,000 queries, we found through manual investigation that 1526 are long extended.

5.2 Query Keywords and Recognizing the Long Extended

In the majority of query log studies, the queries are classified according to keyword to give an illustration of the types of queries being put forwarded. Even though this type of classification is valuable for that point, for our assessment these categories do not facilitate in choosing whether queries are part of the long extended of queries we

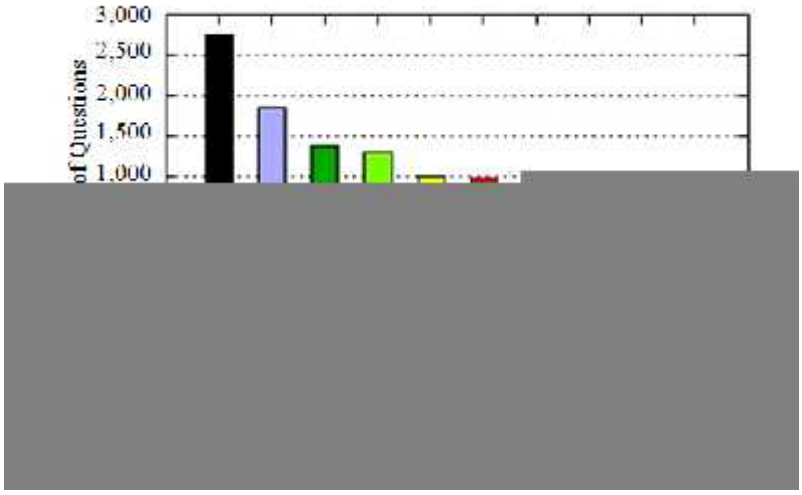


Figure 3: Mahalo number of queries per group as available on Mahalo's website [1].

wish to assess. Figure 3 demonstrates the available queries per group on Mahalo's website [2]. This structure of classification by keyword is similar to earlier studies [28, 29, 39]. From these groups it is ambiguous whether keywords such as "Entertainment" and "Travel" could map to verticals or are excessively extensive and should be directed to our structure. Moreover dividing keywords into sub-keywords (using Mahalo's available breakdowns) exposes that several of these finer granularity of sub-keywords are straightforwardly mappable to verticals: e.g. Yellow Pages" and "Definitions" may be mapped straightforwardly to data sources at "sulekha.com" or "wikipedia.com" with little endeavor. Other sub-keywords are still too extensive for a straightforward vertical (e.g. the sub-keywords of "Health such as "Travel" or "World Governments"). Each query from Mahalo has an related set of keywords it belongs to as allocated by Mahalo. Table 3 lists the top 10 most vastly represented sub-keywords that identified in our illustration (127 keywords total), the proportion of queries belonging to these sub-keywords, and whether there are equivalent verticals that are executed in any existing SMS search methods. The sum of the proportions of the entire table do not add up to 100 because queries may belong to more than one keyword. Certain keywords such as "Celebrities" have online databases such as Internet Movie Database (IMDB) that could effortlessly be used as a resource of structured information to respond queries, and executing verticals on top of these would not be difficult. Using the sub-keywords, we detached queries from our 2,000 query mass that do not have accessible verticals in any existing automated SMS search structure to recognize the long extended queries (1526 long extended queries).

Table 3: Mahalo top sub-keywords by proportion and existence of probable verticals.

Topic	% of total questions	Existing Verticals?

Celebrities	12.4%	No
Yellow Pages	9.8	Yes
Movies (not show times)	10.1	No
Music	9.2	No
Definitions	11.1	Yes
Relationships	8.3	No
Food & Drink	6.5	no

These remaining queries are precisely long extended questions SMSSearch is intended to answer. Several examples are listed in Table 4. The query allocation across sub-keywords and the variety of queries in general proposes that mobile user information requirements even on low-end devices are more varied than formerly demonstrated [29]. This may be due to the more significant input modality of voice or the user's observation of the structure as being extremely intellectual. We reschedule a more concluded relationship between these mobile search patterns and those in the literature to prospective studies.

5.3 Baseline Assessment

We carry out a few baseline assessments of our application using our sample of 2,000 queries from Mahalo containing both vertical and long extended queries. We alter these queries assuming the client understands how to use the application and is prepared to enter keywords along with a clue word rather than a normal language query. For each Mahalo query we rewrite the query exclusively by removing and

rearranging words given that we have understanding of how the application works and be familiar with what would likely be a superior alternative for the clue word. As an example, "what are the symptoms of HIV Infection" is rewritten as "HIV Infection symptoms" and the entire query is submitted to Google with "symptoms" used as the clue word. Using these queries we find that our application (including redirection to available verticals) outcomes in 62.5% correct answers. In contrast, Google SMS returns accurate outcomes for only 9.5%

Table 4: Example queries from Mahalo(together with spelling errors).

Question
Which is the world largest railway station
which of the disciplines is Nobel Prize awarded?
Who was Galileo
First China War was fought between which country
What is the dimensions of basketball court
On which date the World Red Cross and Red Crescent Day is celebrated
Film and TV institute of India is located at which place

Table 5: Outline of SMSSearch outcomes.

Input, Output	% Correct
Mixed, Snippets	62.5%
Long Extended, Snippets	56.4%
Long Extended, N-Grams	23.7%
Long Extended (w/clue), TF-IDF Snippets	19.9%

Long Extended (w/out clue), TF-IDF Snippets	6.1%
Long Extended Unmodified Queries, Snippets	15.9%
Long Extended Unmodified Queries, N-Grams	6.7%

of these queries. We note that the low performance of the Google SMS could be due to a multiplicity of causes that we discuss in Section 7, and the consequence is presented here only for reference.

What is more motivating is if we eliminate the queries that are forwarded to existing verticals. To focus on the core SMSSearch algorithm, we think about only the performance of the SMSSearch algorithm on the 1526 long extended query subset. All additional assessment in this section is carried out with this set of 1526 long extended queries. Table 5 summarizes the outcomes of our assessment. We find that for the long extended queries SMSSearch returns 56.4% correct results. Moreover, if we think about only the uppermost n-grams returned rather than the whole snippet, the performance fall to 23.7%. These outcomes largely specify that the raw performance of SMSSearch has significant room for enhancement, and returning snippet replies usually results in enhanced performance.

We observed before that there are concerns with readability, but what do the snippet outcomes in fact look like? A representative set of examples is shown in Table 6 for both accurate and inaccurate outcomes. Contrasted to the Mahalo individual written replies we examine that the readability of our snippets is poor and could be enhanced; On the other hand, finding the most favorable snippet structure is a separate problem where existing methods could be useful [30].

5.4 Is Refinement Using N-grams Profitable?

Because we are returning snippets rather than n-grams, it is normal to request whether n-grams are essential or if grading snippets alone would execute just as well. To confirm that the middle refinement phase using n-grams is profitable we compare beside a straightforward per-word TF-IDF approach. In

the first part of this trial we collect SMS sized snippets straightforwardly using

a 140 byte sliding window across all documents. In this second part of the trial, we do the same thing excluding the snippets (d_j) are scored based on the summation of the standard TF-IDF

scores as the metric for each of the i words (o_i) of the query.

We find that with merely TF-IDF, the accurate outcome is returned in only 6.1% of queries. The

outcomes of the per-snippet correctness are shown in Table 5 for assessment. With the clue word used in combination with TF-IDF 19.9% of queries are appropriately returned. On the whole, both raw TF-IDF and TF-IDF with a clue word return inferior outcomes than with the middle refinement using n-grams.

Table 6: Example queries with snippet replies returned by our application.

Original Question	**Example Snippet Answer**	**Judged Correct**
How many thief's are there in alibaba tale	There are twenty three tales in the Canterbury Tales.	**"**No**"**
What is software	software:1:(computer science) written programs or procedures or rules and associated documentation pertaining to the operation of a computer system and that are stored in read/write memory.	**"**Yes**"**

Where does humpty dumpty live?	Humpty Dumpty was created by author Lewis Carroll and is in Through the Looking Glass. Therefore, he must live in Wonderland.	"Yes"
In which streams the noble prize is awarded	Nobel Prize in Physic, Chemistry, Medicine and Literature are awarded in Stockholm,Sweden, the Nobel Peace Prize is in Oslo, Norway.	"Yes"
Which is India's first super computer	IBM's Naval Ordnance Research Calculator was the first supercomputer and the most powerful computer on earth from 1954 to 1963.	"No"

5.5 Returning Several Snippets

It is imaginable that for the queries that are not replied well, such as unclear queries or clarifications, somewhat longer snippets or additional outcomes could improve our results. The application may permit several SMS messages to be requested in the form of a "supplementary" link. To investigate this prospect we trialed with returning several snippets. We compare two dissimilar snippet selection techniques to improve outcome multiplicity. In both techniques, we first arrange the snippets by grade. The first technique is to return a snippet for each of the top graded n-gram outcomes. The second

technique returns only snippets for the top graded n-gram outcome, but needs that the returned snippets are from different occurrences of the clue. We find that as more outcomes are returned, the number of queries replied boost slightly (1 - 5%) for each supplementary reply returned.

Both techniques demonstrate a similar rate of enhancement, but the first technique of maximizing n-gram multiplicity constantly performs enhanced by around 10%.

5.6 How Significant is the Clue Word?

One meta query we have considered briefly in our assessment is: can people in fact come up with clues that are helpful in the first position? Requiring an supplementary clue word is "suitable"

because existing applications by Google and Yahoo do precisely this, so our user interface is at least no inferior than the standard. Conversely, it is motivating to see how responsive our application is to the existence of a well specified clue word.

We examine that if the clue word is not appropriately identified, and randomly assigned as the last word in the query, the precision of the snippets drops from 56.4% to 15.9% when compared to the modified queries. Table 5 demonstrates the performance of our algorithm on our queries exclusive of any clue word identification or query modification for evaluation. This recommends that the clue word is significant to our application and a commonly useful method for focusing on the preferred snippets of information.

6. Related Work

There has been comparatively small research on SMS-supported search. In this part, we take somewhat superior vision of the problem space and compare SMSSearch with mobile search applications, query/reply (Q/R) applications from the Text Retrieval Conference (TREC) community [12], and text

summarization systems.

6.1 Mobile Search Features

Mobile search is a primarily different search standard than conventional desktop search. Yet, we carry on to view mobile web search as either an expansion of the desktop search model for high-end phones (such as PDA/iPhone) or somewhat limited version through XHTML/WAP on low-end devices. Mobile search in both of these settings be different from conventional desktop search in a number of ways as revealed in latest studies by Kamvar et al. [28, 29]. The first study [28] found that the mobile search click-through rate and the search page views per query were both significantly inferior in contrast to desktop search. Implication, most mobile search clients be inclined to use the search application for short time-periods and are either satisfied with the search engine snippet replies or do not find what they were searching for. The study also establish that the determination of mobile users was incredibly low signifying that the huge mass of mobile searchers move toward queries with a specific keyword in mind and their search frequently does not direct to investigation. The second study [29] showed that the multiplicity of search keywords for low-end phone clients was a lot fewer than that of desktop or iPhone-based search. This outcome recommends that the information requirements are extensive, but are not satisfied by the information applications available on low-end phones.

We find supported indications in our analysis of Mahalo queries that mobile query multiplicity

across keywords is elevated given a more easy-to-read input modality such as voice. As a whole, these studies signify a imperative need for rethinking the existing mobile search model for low-end mobile devices.

6.2 SMS-supported Search Applications

SMS-supported search is very different from conventional mobile search via XHTML/WAP. A striking

characteristic of SMS-supported search is the inferior barrier to entry of SMS (in contrast to other data applications) due to the use of low-end phones and extensive availability of SMS. In current scenario , SMS is the most omnipresent protocol for information exchange next to voice. In addition, there is assumption that "the financial recession will most likely reduce expansion for the more luxury-based mobile applications, but SMS is estimated to maintain its escalation as it is popular, economical, reliable and confidential." [4]. A lot of the top search engine companies like Google [6], Yahoo! [16], and Microsoft [15] have entered the SMS search marketplace and developed their personal versions of SMS-supported search applications. All these applications have been customized for very specific keywords (e.g. directory, weather, stock quotes) and specific in nature. These computerized applications are not the only ones available. Mahalo [1], Just Dial [7] (in India), and Google Baraza (in Africa) [5] are SMS- or voice-based query/reply applications using a person to respond to queries. The queries to Mahalo and JustDial are in normal language query form, understudied by a an individual who looks for the answer online, and responds with a reply via an SMS. A latest confidential study by mSearchGroove [10] comparing the precision of replies of these applications has revealed that the computerized applications experienced from low precision when fulfilling a random set of queries: Google SMS 22.2%, Yahoo! oneSearch 27.8%, as compared to 88.9% for Mahalo. However, the study also observed that the middle response time for the computerized applications was on the order of 10-14.5 seconds, whereas the middle response time for Mahalo was 227.5 seconds. Linking an individual in the loop significantly increases response time.

The problem we seek to reply is how do we construct an application that attains the best of both worlds: an SMS search application that is both a rapid (automatic) and provides precise query replies? One cause this problem is inflexible is that search queries are naturally vague, yet returning a disambiguated reply is particularly essential to SMS search queries for various reasons [37].

6.3 Query/Reply Applications

The difficulty that SMSSearch seeks to deal with is similar to, but different from, conventional query/reply applications developed by the Text Retreival Conference (TREC) [27] community.

TREC has developed into many separate tracks dedicated to specific areas such as: simple ad-hoc

tasks, query/reply tasks, million query tasks, etc. Nonetheless, our problem is dissimilar from each of the TREC tracks for at least one of three dimensions: (a) the character of the input query; (b) the document compilation set; (c) the character of the search outcomes in the query reply. These three dimensions are derived by Carmel et. al. [20] in a representation for assessing query difficulty.

First, from the input query point of view, SMSSearch is mainly suitable for unstructured search

style queries. SMSSearch can be extended for straightforward normal language style queries using existing query alteration methods as we discuss shortly. The allocation of the queries is also reliant on the mobile framework (i.e. SMS or voice queries as conflicting to Desktop search queries or iPhone queries). Second, SMSSearch is a snippet mining and snippet grading algorithm that outputs reduced text snippets in existing pages as search replies. Third, the most remarkable differences of our problem in contrast to many TREC tracks is that in SMSSearch the compilation of documents being searched over is a set of diverse, “noisy”, and hyper-linked documents (web pages) indexed by Google. In contrast, the TREC ad-hoc track and TREC query/reply tracks have until recently used a set of newswire documents that are very spotless, and blog documents that have some noise [25]. Newswire and blog sets are also not hyper-linked and are quite a few orders of magnitude lesser than the web. Previous work suggests that the extent of the set influences how well link examination methods may be helpful to information retrieval [36]. Further recently the TREC web, terabyte [23], and million query [17] tracks have employed hundreds of millions of web pages as the document set to permit for link analysis methods to be applied

toward the tasks, and there have been many applications that influence the web for either the main or supplementary mass. The truth that the set is at least two orders of magnitude larger than any TREC assessment, significantly noisier inwords of information multiplicity, formatting, and being hyper-linked means that it is a dissimilar (and more demanding) problem, which directs us to implement a different elucidation. The initial and

most intimately related application to SMSSearch in words of construction and approach is AskMSR [19]. AskMSR also leveraged the data redundancy of the web as contrasting to complicated linguistic methods to mine n-gram answers. SMSSearch shares methods with earlier applications [19, 22, 38], but is intended specifically for the exclusive necessities of SMS search in words of its input and output. SMSSearch anticipates mobile search queries along with a clue and returns snippets whereas existing applications may expect normal language queries and return n-grams or whole paragraphs and perhaps the source document.

6.4 Computerized Text Summarization

Snippet mining is also imaginatively associated to summarization, but only to the mining task (finding sections of the text and creating them precisely), and not the generalization task (creating material in a new way). Though, the main dissimilarity is the objective of these two problems. The objective of mining is still to summarize the contents of one or more documents. The objective of snippet mining in SMSSearch is to find the accurate reply to a query. For this reason, our problem is more alike to information retrieval. That said, many of the methods are used in both areas of research together with: Naive-Bayes methods [18], Rich features [33], and other statistical techniques.

7. Debate

The blueprint of our application and the assessment was motivated by the SMS search problem. Our

objective was to recognize how existing techniques would execute in this problem domain and not to succeed or reinvent the huge amounts of research in the Information Retrieval (IR) space.

7.1 Data Collections

We had primarily trialed with Google Trends queries and other desktop search logs, but found that the majority of queries were too vague and did not match the type of queries found in mobile environments. We determined that the Mahalo data collection would supply more practicality as to the authentic queries people would make. To get a logic of the query difficulty and confirm that the Mahalo queries are in fact "accuracy oriented" we performed a superficial assessment based on work by Mothe and Tanguy [35] to assess the Syntactic Links Span and Average Polysemy Value of the queries in our mass compared to those used in a variety of TREC tracks. We confirm that for these two features our mobile queries are significantly dissimilar from those in the query/reply track. The closest counterpart for these two features was the set of "search query" layout queries from the Million Query Track.

7.2 Difficult Nature of Queries

In terminology of the presentation of our application, we observed that the queries that are not replied appropriately despite of layout are frequently vague, elucidations, enumerations, problems that need scrutiny, or time-sensitive queries. This is not unexpected as many statistical techniques have alike restrictions. Our algorithm in particular needs proximity to the clue word and we also anticipate the reply to be only a few words long. Queries those are vague such as:

"how many calories are in hot dog" are ambiguous in the sense that the human aptitude was essential to suppose that "Who cooked the hot is nearly equivalent to "hot dog". *Elucidations* are probable to need more space to reply appropriately. We examine that queries such as "how do i install an operating system for my PC? i have a Windows 7 CD" are unlikely to be replied within 140 bytes by still a human

aptitude. *Enumerations* such as "words that rhyme with long" are also difficult for SMSSearch because the reply is doubtful to all emerge successively near any clue word in a significant number of places on web pages. *Scrutiny* queries (e.g. "what is the least expensive country to live in") are challenging since the reply is not instantaneously available in the text of web pages. *Time sensitive* information is difficult for a mixture of need of web page sources and closeness to a valuable clue word. Given the ubiquity of these difficult query types, it is rather unanticipated that our straightforward algorithm is capable to respond over half of the data collection.

7.3 Contrast to Existing Applications

The enclosure of Google SMS performance on our data collection is only to show that the information requirements of people using only voice and SMS to ask queries may not be detained by that particular apllication of verticals. A direct contrast between the two numbers is irrational because we have no information as to the allocation of queries Google SMS receives and how that compares to our test collection. It is completely likely that the allocation of Google SMS queries is dissimilar, and their verticals effectively convince a large proportion of queries acknowledged by their service.

7.4 NLP and Mobile Craigslist

Merging statistical and linguistic methods in the perspective of replying queries using the web has been comprehensively surveyed by Lin et. al. [34]. As part of future work, we plan to integrate more refined statistical and Natural Language Processing (NLP) methods. Specifically within the NLP summarization literature, there are probabilistic representations that discover how significant a range of features of a given gravel are for text summarization [18]. One primary difference in SMSSearch in contrast to NLP methods is that the mass varies asa function of the search query; given a query, the mass for SMSSearch is the search pages returned by a search engine. Given a extremely inconsistent mass, straightforwardly applying existing NLP-based probabilistic models may not be suitable since the model should be a

function of the query. To better appreciate this contrast, we briefly illustrate the problem of SMS-supported Mobile Craigslist where the objective is to make available SMS-supported search responses for Craigslist queries. This is a problem we are presently investigating where probabilistic models derived from NLP are straightforwardly appropriate since there exists a well-defined mass for any given search group (e.g. automobile, tourism, real estate). Using this mass, we can use standard NLP techniques to be trained the various features for each group and obtain a concise summary of each Craigslist posting for a group based on the features. Having a well-defined mass considerably helps in enhancing the precision of summarization.

8. Deployment

We install our application with a small focus set of 30 clients consisting of both urban peoples and college students in Meerut, India. Our trial lasted for three weeks, during which, clients with low-end mobile phones and no data plan found the application to be the most valuable. Verticals were not primarily employed in our trial, and SMSSearch was tasked with responding all queries. We found in our preliminary response session that people requested local business contact information and other straightforward verticals that were not being replied properly by SMSSearch. To concentrate on this, we deployed a number of verticals to enhance query exposure. After the enclosure of verticals with specific keywords, we found that our clients took time regulating to the sentence structure of requiring the clue/keyword at a specific location of the query. We are presently exploring probable enhancements to both the client interface and performance, and we suppose to enlarge our level of use over the next few months.

9. Conclusion

In this paper, we have demonstrated SMSSearch, an computerized SMS-supported search response

application that is customized to work across random keywords. We find that a mixture of straightforward Information Retrieval algorithms in combination with existing search engines can make available practically precise search responses for SMS queries. Using queries across random keywords from a real-world SMS query/reply service with human-in-the-loop responses, we demonstrate that SMSSearch is capable to answer 62.5% of the queries in our test collection. We also demonstrate how SMSSearch is integrated into an overall SMS-supported search application, and reveal its applicability in an open public beta in Meerut, India. Although more influential Information Retrieval and Natural Language Processing techniques are bound to enhance performance, this work correspond to a venture into an open and realistic research domain. In addition, our techniques may be realistic to other constrained Information Retrieval tasks over unstructured documents away from SMS-supported web search.

REFERENCES

[1] Mahalo. http://www.Mahalo.com.

[2] Mahalo categories. http://www.Mahalo.com.

[3] Esoko. http://www.esoko.com/.

[4] Global - key telecoms, mobile and broadband statistics. http://www.budde.com.au/.

[5] Google baraza. http://www.google.com/baraza/.

[6] Google sms. http://www.google.com/sms.

[7] Just dial. http://www.justdial.com/.

[8] Kannel. http://www.kannel.org/.

[9] Linguistic data consortium. http://www.ldc.upenn.edu.

[10] Pump up the volume: An assessment of voice-enabled web search on the iphone. http://www.mcubedigital.com/msearchgroove/.

[11] Sms gupshup. http://www.smsgupshup.com/.

[12] Text retrieval conference. http://trec.nist.gov/.

[13] Trec question answering track. http://trec.nist.gov/data/qamain.html.

[14] Twitter. http://www.twitter.com/.

[15] Windows live mobile. http://home.mobile.live.com/.

[16] Yahoo one search. http://mobile.yahoo.com/onesearch.

[17] J. Allan. Million query track 2007 overview. Technical report, DTIC Document, 2007.

[18] C. Aone, M. Okurowski, and J. Gorlinsky. Trainable, scalable summarization using robust NLP and machine learning. *Proceedings of the 17th International Conference on Computational linguistics-Volume 1*, 1998.

[19] E. Brill, S. Dumais, and M. Banko. An analysis of the AskMSR question-answering system. *Proceedings of the ACL-02 Conference on Empirical methods in natural language processing-Volume 10*, 2002.

[20] D. Carmel, E. Yom-Tov, A. Darlow, and D. Pelleg. What makes a query difficult? *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2006.

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

[22] C. Clarke, G. Cormack, G. Kemkes, M. Laszlo, T. Lynam, E. Terra, and P. Tilker. Statistical selection of exact answers (MultiText experiments for TREC 2002). *Proceedings of TREC*, 2002.

[23] C. Clarke, N. Craswell, and I. Soboroff. Overview of the TREC 2004 terabyte track. *Proceedings of TREC*, 2004.

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

[25] H. Dang, D. Kelly, and J. Lin. Overview of the TREC 2007 question answering track. *Proceedings of TREC*, 2007.

[26] 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), 2002.

[27] D. Harman. Overview of the first text retrieval Conference (TREC-1). *First Text Retrieval Conference (Trec-1): Proceedings*, 1993.

[28] M. Kamvar and S. Baluja. A large scale study of wireless search behavior: Google mobile search. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2006.

[29] M. Kamvar, M. Kellar, R. Patel, and Y. Xu. Computers and iphones and mobile phones, oh my!: a logs-based comparison of search users on different devices. *Proceedings of the 18th International Conference on World Wide Web*, 2009.

