The Role of Context in Query Input: Using contextual signals to complete queries on mobile devices

Maryam Kamvar

Google Inc., Mountain View, CA
Columbia University, New York, NY
mkamvar@google.com

Shumeet Baluja

Google Inc.,
Mountain View, CA
shumeet@google.com

ABSTRACT

The difficulty of entering queries from impoverished keyboards impedes the use of web search on mobile devices. On average, it takes a mobile user approximately 60 seconds to enter a query from a 9-key keypad [1]. In this paper, we explore the use of contextual signals to facilitate query entry on mobile phones. We present a query prediction system which offers automatically generated word completions as the user is typing her query. The query prediction system redefines the prediction dictionary after considering contextual signals such as the application being used (e.g. search vs. general text messaging), the inferred location of the user, the time of day and day of week. We demonstrate a 46.4% improvement in query entry, measured by number of key presses needed to enter queries. We found that the two contextual signals that make the largest impact are knowledge of the application being used and the location of the user.

Categories and Subject Descriptors
H.5.5 [Information interfaces and presentation] (e.g., HCI): Hypertext/Hypermedia

General Terms
Experimentation, Measurement, Human Factors

Keywords
Mobile search, query, word completion, context, web search, query prediction

1. INTRODUCTION

As internet access becomes available from an increasing variety of places, times and devices, the need for understanding the context from which a user is interacting with information will grow. Incorporating a user's context in web search has been shown to improve the quality of search results returned to the user in several ways: by directing the level of granularity of the results that are returned, by influencing the sort order of those results and by helping format them. In this paper, we show the benefit of considering context *before* the user issues a query. We explore the use of context with the goal of facilitating query entry through a query prediction system. Although our system can be used in conjunction with any typing interface, we focus our attention to the mobile search environment where it can have the largest impact on the user experience.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Conference'04, Month 1–2, 2004, City, State, Country.
Copyright 2004 ACM 1-58113-000-0/00/0004…$5.00.

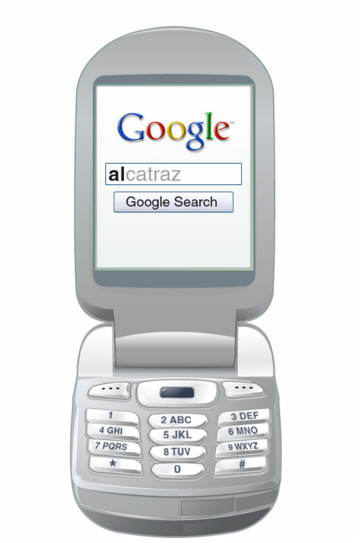

Figure 1: An example query prediction interface used with Google mobile search: The user has typed in the letters "al", and after considering her location (that she is in San Francisco, CA) we suggest the most probable completion of the prefix: "alcatraz" (shown in faded grey)

To perform a search on typical mobile devices, users must enter their query using a 9-key keypad. These small keyboards have been shown to hinder query entry; it takes a mobile user approximately 60 seconds to enter the average mobile query from a 9-key keypad [1]. The average mobile query is 2.3 words and 15.5 characters long [1]. Our goal is to reduce the time to enter a query by requiring the user to enter fewer letters per query.

The system presented in this paper reduces the key presses needed to enter a query by offering word completions as the user is typing. If the suggested completion is correct, the user can accept it with a single key press; if it is not correct, the user can continue typing as normal. To generate the completion, the system considers contextual signals such as location, time, and day of week. Additionally, we examine the use of a contextual signal often overlooked – the context of the task being accomplished; which in this case is that of typing search queries. An example scenario is shown in Figure 1: the user has typed in the letters "al", and after considering her location (that she is in San Francisco, CA) we suggest the most probable completion (shown in faded grey): "alcatraz".

In order to gain an understanding of which contextual signals are useful in improving query prediction, we built several query prediction models. Each model incorporates a different contextual signal. We find that the two contextual signals that make the biggest impact in reducing the number of key presses needed to enter a query are knowledge of the application being used (in this case a search engine), and the location of the user. When combining these signals, we obtain a 46.4% improvement over having no text prediction system, and a 34.9% improvement over the standard dictionary-based word prediction systems. Signals such as hour of day, day of week and phone carrier were found to be less effective in improving prediction quality.

The remainder of the paper is organized as follows: In Section 2, we provide an overview of related work in two areas: research in the use of context to improve search systems and existing mobile text entry systems. Section 3 describes the algorithm used to incorporate context in the query prediction system. Section 4 describes our dataset, the setup of the query prediction experiments, and the metrics used to measure the improvement of each prediction model. Section 5 presents the results of our experiments. In Section 6, we discuss the impact of different user interfaces to our prediction models. We close the paper in Section 7 with conclusions and suggestions for future research.

2. RELATED WORK

The use of context in information retrieval systems has been extensively studied; for an overview of the issues and literature, the reader is referred to [2][3]. There have been many types of context considered and numerous methods through which contextual information has been used. For example, [4] looked at a user's past behavior to determine attributes about the user's knowledge of a topic in order to determine how to tailor the information that is retrieved. [5] describes early work in applying simple contextual clues to the types of results returned with web search engines, such as restricting the results to specific categories, etc. An alternate approach to capturing context on the web is proposed by Finkelstein[6]; here, a user marks a particular place in a document pertaining to a subject that they would like more information. In Finkelstein's system, context refers to the surrounding words, and is used to bias the types of results returned in response to the information request.

Other work has attempted to capture physical attributes about a user's context. [7] explores the impact on the search process when the user is in an environment in which she faces many interruptions. In work more closely related to the study presented here, [8] presents information retrieval and filtering for mobile device access, and discusses the role of geo-locating a user, for example with the use of a global-positioning-system device (GPS). They also consider the modification to the human-computer interactions that will be required to support contextual information retrieval and discuss methods of presenting the results to the user.

Placing our work within the framework of these studies, we attempt to move the role of context earlier in the search process. We assert that context is useful before the query is entered.

It should be noted that there are a variety of systems that attempt to decrease the time users spend inputting text on their mobile phone. The systems such as eZiType can be classified as word prediction systems as they complete the word before all the letters are pressed, and systems such as T9 (in its common instantiation) are word disambiguation systems[1]. iTap is a hybrid which has both word disambiguation and word prediction capabilities. A more detailed analysis of each of these systems can be found in [9]. Our system will work with any of the text entry approaches; it will not only improve the speed of the text entry, but also improve the accuracy of the word-disambiguation that systems such as T9 use.

The core components of each of the above-listed systems are the user interface and the dictionary which defines the probability that a word will occur. We employ a simple interface which displays the suggested word completion in-line as the user is typing. Our dictionary is dynamic – the probabilities associated with each word change based on the user's context.

3. ALGORITHM

To generate a word prediction, we find the word with the appropriate prefix which has the highest probability of occurrence in the set of available words. If we choose to incorporate a contextual signal in the prediction model, we further restrict the set of words to those which also hold the signal's value. We can represent each word's probability of occurrence as `Probability(word | prefix & context)`. If the contextual restrict has yielded an empty set of words (if the probability of all words in the set are equal to 0) we lessen the restriction.

Essentially, for each context (or signal value) we redefine the vocabulary over which the word prediction is generated. For example, if we are considering the location of the user as context of the query, the signal value can be any city in the US. If the user was querying from San Francisco, California, we would restrict the set of words considered for auto-completion to those prior queries which were also made in San Francisco. If the set of available words is empty, for example if there were no queries made from San Francisco, we expand the set of available words by lessening the restrictions – we might allow queries made anywhere in California to be considered.

4. EXPERIMENTS

In order to gain an understanding of which contextual signals are useful in improving query prediction, we built several query

[1] The difference between word prediction and word-disambiguation systems should be noted. Word disambiguation systems interpret the key presses as the most likely word (i.e. typing "73776625492846" is equivalent to typing "personalization"). Word prediction systems try to predict the word "personalization" before all the keys are pressed. The latter (word prediction) is the focus of this paper.

prediction models. Each prediction model incorporates a different contextual signal, and we tested the model's accuracy on a set of 1 million queries entered by Google users. We had two different flavors of each model – one completed a single word at a time, and the other could offer a query-based, or multiple-word completion.

This section first provides a description of the dataset used for both training and testing the query prediction models. Section 4.2 provides an overview of our experimental setup and in Section 4.3 details the metrics used to measure the effectiveness our approaches.

4.1 Dataset

The data set from which context is determined consists of over 6 million randomly sampled incoming queries sent to Google via Google's SMS search service. A complete description of the SMS service is provided in Section 4.1.1. The queries are taken from an 8 month period in 2006 and are sampled from the top 1000 query-generating cities in the US. All of our data is strictly anonymous; we maintain no data to identify a user. All of the results we report are aggregate statistics.

Each SMS request includes the user's query, the user's carrier (mobile service provider), a timestamp, the number of results returned to the user, and the classification of the query to a query type (sports scores, weather query, local listing etc).

Only queries originating from US carriers with one or more returned search results were considered. Although valid SMS queries include stock symbols, zip codes, movies, etc.[2], we restricted the queries under consideration to be those which are for local listings. Local listings are not only directory-type searches, such as for exact businesses (*dominos 94114, Walmart Washington DC)*, but can also be category searches – eg "*pizza 94114"*, "*sunglasses, Portland,Oregon").*

The location information is explicitly included in the query by the user; no cell-id or GPS information is associated with a query. From the set of 6 million queries, we extracted the query's location from the query. For example, if the query was "pizza 10023" the query term would be "pizza" and the location term would be "10023". Similarly, if the query was "pizza San Francisco CA" the query term would be "pizza" and the location terms would be "San Francisco CA". Our experiments try to predict only the query terms, not the location terms. The location terms of each query were translated into a city, county, and state vector and used as a proxy for the user's current location.[3] Although there is no strict guarantee that the location terms correspond to the location of the user, we speculate that for most mobile searchers their location of interest and physical location are the same.[4] For the remainder of this paper, when we refer to query terms, we refer to the location-independent terms in the query. When we refer to the location of the SMS user, we refer to the city, county, state vector generated from the location-terms in the query.

In order to ensure that our system works well when deployed, the testing set is extensive and completely independent from the training set. Our training set was gathered from randomly sampled queries from January-August, 2006 and we test our models on a randomly sampled set of approximately 1 million queries made in September 2006. The use of an independent set ensures that we do not over-fit our models to the training set, and the use of time-ordered testing and training ensures a realistic implementation scheme.

It should be noted that the query word frequencies over the 9 month period for which we collected data are relatively stable for SMS queries. There were no significant differences in the results presented from the results generated when we trained on January data and tested on February data, and when we trained on January through September data and tested on October data.

4.1.1 Google SMS Interface: Background

For the reader who is not familiar with Google's SMS-search service, in this section, we provide a brief introduction to its features. To perform a Google search via SMS, users in the U.S. perform the following steps:

1. Start a new text message and type in the search query
2. Send the message to the number "46645" (GOOGL)
3. The user will receive an incoming SMS from Google with the results of her query. Long results may span multiple SMS messages.

The format of the search results returned to the user is specific to the query type. The results presented for a local search are the top 3 of those on the desktop (HTML) interface at http://local.google.com. If the query is for weather, the user will get the 5 day weather forecast for the location she has specified. If the user queries a stock symbol, the current stock price would be returned along with its open, high, low and average volume numbers. The full list of query types supported by SMS search is listed at www.google.com/sms. As previously mentioned, in our study we only consider SMS queries which are for local listings.

4.2 Experimental Setup

A large number of experiments will be presented in this paper. In each experiment, we attempt to predict each of the 1 million queries made in September, knowing as few letters of the query as possible. We evaluated the performance of both word-based prediction models and query-based or multiple-word prediction models for each contextual signal. The word-based prediction

[2] The full list of SMS features can be found at http://www.google.com/sms

[3] The process of extracting a location from a query and converting it to a consistent geo-data structure was developed elsewhere at Google and is beyond the scope of this paper.

[4] Although multiple means of mobile geo-location have been developed (e.g. Global Positioning System, GPS), there has not yet been widespread integration of this technology with search services. Thus, in this study, we must rely on location cues explicitly entered by the user. Looking forward, for the majority of queries, integration of GPS will ease the user's burden of explicitly specifying her location of interest; the user will only need to explicitly specify her location of interest if it is different than her physical location.

models isolate the effects of the contextual signals; in the query-based models, we exploit the knowledge of common multi-word queries to further improve the query prediction system.

4.2.1 Word-based Model

Each query was considered on a word-by-word basis. The model generates a word prediction based on the prefix of the word that had been entered. If the prediction was the same as the intended word, the word is completed by accepting the prediction (this is counted as one keystroke). If it was not correct, the next letter of the word would be appended to the prefix, and the word prediction would be recomputed based on the new, larger, prefix.

The *maximum* number of key presses per word is equal to the number of key presses required to enter the word if there was no word completion interface available (on QWERTY keyboard, this would be equal to the length of the word). The *minimum* number of key presses per word is two[5], because we cannot offer word completions until the user starts typing and the user must also expend 1 key press accepting the suggested word.

The first set of experiments assumed query entry on a 9-key keypad. We assumed that it took 3 key presses to enter a non alpha-character (numbers and symbols). We also assigned a 0.5 key press penalty for entering consecutive letters on the same key. For example, according to our calculations the word "bat" would take 4.5 keystrokes – "b" requires two key presses, "a" requires one key press and "t" requires one key press. To enter the "a" after the "b", the user would either have to press the "next" key or wait for a the multi-tap to timeout, thus incurring a small penalty.

Below, we examine a simple example, where the intended query is "apple farm":

1. Since a user must trigger the system by entering the first letter of the query, the model attempts to predict a word based on its first letter; in this case the letter "a". Imagine that the word with the highest probability to complete the prefix "a" is "artist". This is the word the user may choose to complete her prefix.
2. However, "artist" is not the correct prediction, so the next letter of the intended word is appended to the prefix. This is equivalent to the user ignoring the suggested word completion, and continuing to type the next letter of her intended word. A "p" is appended to the existing prefix, and the model attempts to predict the most likely completion to the prefix "ap".
3. At this point the top suggestion is "apple", and since it is the intended word, it is accepted. We assume a user will always accept a correct word suggestion.
4. Thus the number of key presses to enter the word "apple" using this particular model is 3: one "a", one "p" and one "select-completion-key".
5. When a suggestion is accepted, a space is automatically appended to the word.
6. "Farm" is the next word to predict, and the model suggests a completion for the prefix "f". No word completion will be suggested before the user types the first letter of the word.
7. In this case, the hypothetical system generates "farm" as the word prediction, so the suggestion is accepted.
8. The word farm is completed in 4 keystrokes, one "f" (which requires 3 keystrokes on a 9-key keypad) and one "select-completion-key".

Using our hypothetical query prediction model, the query "apple farm" requires seven key presses to enter, as opposed to 17.5 key presses if no word prediction system was available. Therefore, the average key presses saved per word predicted is 5.25. Each of our experiments use a different prediction model to determine the appropriate completion for the prefix.

4.2.2 Query-based Model

The query-based model is an extension of the word based model. In this system, if the user accepted a word completion, the query completion is then automatically suggested to the user. The user could choose to ignore the query completion, and continue typing the next word in the query as normal, or accept the query completion with an additional "select" press.

In the "apple farm" example outlined above, steps 1-4 would be exactly the same but would continue:

5. When a suggestion is accepted a space is automatically appended to the word, and a query completion is generated.
6. In this hypothetical system, the most likely query that starts with the word "apple" is "apple farm" so "farm" is suggested as the query completion.
7. The user accepts the query completion with a single "select" press.

Using the query-based completion model, the query "apple farm" requires only four key presses to complete; three key presses to enter and accept the word "apple" and one key press to accept the query completion "farm".

4.3 Performance Measurements

To measure the improvement of each model, we use three metrics:

- Percent decrease in the number of key presses required to enter the set of September queries relative to the key presses needed to enter the same set of queries without the aid of any word prediction system.
- Percent of test words successfully predicted by model. There are two reasons why the model may not predict a word; first, the word may not exist in the training set. We are limited to predicting known words. Second, even if the test word exists in the database, it may not be the top completion and therefore will not be presented to the user. For example, lets examine the test word "car". If the top word for "c" is "cone", the top word for "ca" is "cat", the word "car" will not have been successfully predicted by the time it is fully entered.
- Average key presses saved per word predicted.

[5] Excluding of course, one-letter words.

5. RESULTS

Our experiments were designed to measure the improvement of query prediction using contextual signals. Section 5.1 discusses the improvement gained by considering the application in use. This signal yielded the highest improvement rate. In Section 5.2, we discuss the results of the query prediction systems which consider the location of user in addition to the application in use. Section 5.3 presents the results of the experiments where we considered the query-specific, but non location-based, signals of time-of-day, day of week, and cell-phone carrier. Section 5.4 summarizes the results.

5.1 Taking into account the task: querying vs. messaging

We find that the language of search queries is fundamentally different than the language used for mobile messaging. Current mobile text entry systems are tailored to mobile messaging and use the same dictionary across all mobile text entry tasks. These systems perform poorly for tasks such as query entry.

In this experiment, we measure the performance of a word prediction model which uses a standard dictionary and show a 21.9% improvement if we simply replace the standard dictionary with a task-specific dictionary. Although no mobile text prediction company would release to us their proprietary dictionary, we simulate the standard dictionaries by using the word frequencies found in the British National Corpus (BNC). The British National Corpus is a 100 million word collection of samples of written and spoken language from a wide range of sources, designed to represent a wide cross-section of current British English, both spoken and written [10].[6]

The test set for all experiments was the set of ~1 million September 2006 SMS queries. There were a total of 1,803,470 words and 10,488,231 characters (including spaces between query words) in the test set. Assuming multi-tap input from a 9-key keypad the query set would require 22,835,152 key presses to enter. On average each query had 2.0 words (median = 2, standard deviation = 1.0) and each word had 5.3 letters (median = 5, standard deviation = 2.3)

Using the word frequencies generated from the British National Corpus to train our word prediction model, the number of key presses needed to enter the test set of queries decreased to 18,797,938; a 17.7% improvement over having no text prediction system. However, only 51.3% of all query words were successfully predicted although the vast majority – 86.9% of query words were present in the BNC. Furthermore, the key press savings for individual words was consistently small; on average, a word was predicted when 2.6 characters remained in the word (an average savings of 5.4 key presses per word) and only 28.5% of words were completed with three or more remaining letters (distribution shown in Figure 2). By all our measures, using the BNC word frequencies to predict queries is minimally effective. Unfortunately we believe this approximates the performance of existing word prediction systems that are based on static language dictionaries for entering search queries.

[6] Many text prediction systems have enhanced their dictionaries with emoticons (e.g. ";)" to represent the "wink and smile" emotion) and common slang (e.g. "l8tr" for later). Their omission from the BNC does not pose a significant problem because communication slang and emoticons are not used widely in mobile queries.

In considering the task-based context of the text entry we made our first improvement to the query prediction system by replacing the general language corpus (BNC) with a set of words which is more representative of mobile search queries. We train our prediction models using a *p*ast *q*uery *c*orpus (PQC) which contains 5 million queries gathered from Google's SMS logs from January-August, 2006. The query diversity of SMS local search is lower than that of mobile web search [1]. Our training set consists of 10.5 million words, with approximately 200,000 unique words. The top word accounts for over 2% of overall query volume and the top 10% of words accounted for well over 50% of overall query volume.

We observed a significant decrease in the number of key presses needed to enter the test set of queries by retraining our prediction model on a task-specific corpus. 14,683,374 key presses were needed to enter the test queries using the PQC. This is a 21.9% improvement over the BNC model and a 35.7% improvement over having no text prediction system.

Additionally there was a dramatic increase in the percent of query words successfully predicted to 75.2%. The percent of query words present in the PQC did also increase, but at a rate lower than the improvement; the PQC contained 98.9% of test words. On average, the letters saved per word predicted increased to 3.2 (which translates to an average of 7 key press savings per word) with the median improving significantly. 49.3% of words were predicted with 3 or more letters remaining (distribution shown in figure 2).

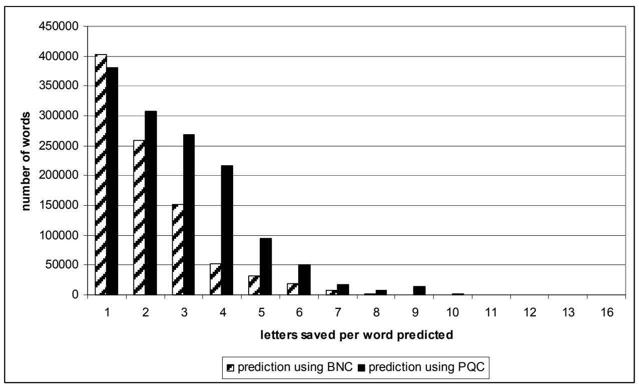

Figure 2: Number of words predicted, grouped by letters saved per predicted word.

Figure 3 is a histogram of average key presses saved for words of a particular length. With both models, the longer the word, the more key presses are saved. It is interesting to note that the PQC model outperforms the BNC model in letters saved for all word lengths except where word length equals 3 and 18. Perhaps this is because the common 3 letter words in search queries (and, the) are better represented in by the BNC word frequencies. We attribute the BNC improvement in 18 letter words to the sparsity of words at that length in both the training at testing sets.

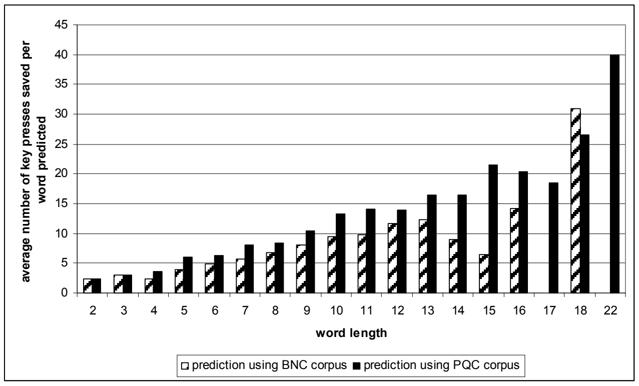

Figure 3: Number of words completed, grouped by number of letters saved

If we consider the performance of the PQC model with query completion (versus single word completion), we see a further improvement: up to a 40.9% reduction in key presses over having no text prediction system in place. Of the query completions that were offered, 34% were accurate completions. The tradeoff between the improvement in query entry and interface clutter added to the user interface should be evaluated through user studies. This is left for future work.

5.2 Location Based Signals

In this section, we examine the utility of location based signals for query prediction. Location-based services (LBS) are often touted as the "killer" applications for mobile devices. There is strong evidence indicating that location-based searches are popular among mobile searchers: a study of Google's 2005 mobile search logs found that local searches were the 4th most popular query category on traditional phones, and accounts for the majority of searches on PDA devices [1].

The appeal of mobile LBS is the potential to automatically apply knowledge of the user's current location to the particular application. There is ample anecdotal evidence in Google's query logs to support the claim that location influences a query. For example, in El Paso, Texas, movie goers are much more likely to query for the term "cinemark" than they are "amc" which seems to be the theatre of choice in most major cities. In Sante Fe, Arizona and Springfield, Missouri, queries for "sushi" are 1000 times less likely as in New York City, San Francisco or Los Angeles. The query "Alcatraz" is queried with much higher probability in San Francisco, CA than any other city in the US. Similar examples can be found throughout the logs; this provides an indication that location-based signals will further improve our system.

In the location based model, we combine the application knowledge (PQC) with information gained by the location signal. We use the PQC, but modify the probabilities associated with each word based on the value of the location signal. In order to take into account a user's location, we created a model that combined the individual probabilities a word would occur in a city, county and state and country in a linear function; we essentially created a geo-spatial smoothing function.

The model generates the top word prediction by weighting the word probabilities generated from each location level: A*Probability(word | prefix & city) + B*Probability(word | prefix & county) + C*Probability(word | prefix & state) + D*Probability(word |prefix). To determine the coefficients for each probability, we performed a linear regression on the data.

Using the location signal in the word-based prediction model gave us an overall improvement of 40.1% over having no text prediction; a 6.8% improvement over using only prefix-based frequencies from the PQC. The average key presses saved per word predicted was 7.5 and the percent of words predicted 78.3%.

To further verify that we did not over-train our model, we also predicted approximately 1 million queries made in October. The performance measurements were very similar (a 39.7% improvement over no prediction, and 6.5 % improvement over using only prefix-based frequencies from the PQC)

The query-based model presents a further improvement to 46.4% improvement over having no text prediction system, and the success rate of query completion rose to 40.3%.

5.3 Non Location Based Signals

Knowledge of the user's carrier, the hour and day of week she was querying did not improve the query prediction models. As shown in Table 5, no significant improvement was gained by considering these signals. There was low variation in performance across each carrier, hour of day and day of week.

In addition to considering each of these signals alone, we created smoothed models for the hour and day signals. For the smoothed hour model, we looked at the probability the word occurred in a specified hour and if the specified hour was in the morning (before noon but after midnight) we evaluate the probability that the word occurred in the morning; otherwise we evaluate the probability that the word occurred in the evening. We expressed this as `P(word | hour) = A*P(word | hour) + B*P(word | am_pm) + C*P(word)`, with the coefficients defined after performing a linear regression on the data. However we did not see an improvement with this model; this is due to the fact that the query diversity did not reduce significantly when we split the set into morning queries and evening queries.

We also combined these signals with location; for example we considered the combination of the user's location and hour of day. Again, we determined the coefficients through linear regression. However, we did not see any improvement. In fact, there was a degradation in performance. We believe the degradation is due to the sparsity of information created by the number of data segments required for using these models.

5.4 Summary of Results

Table 1 summarizes the effect of each contextual signal on query prediction. We improved query entry by 46.4% when taking into account the contextual signals of user location and application in use. To provide a baseline comparison to current word prediction systems, the first row presents results for word-based query prediction when no signals were considered, as is the case in standard word prediction systems. We simulate the standard dictionaries by using the word frequencies found in the British National Corpus (BNC). Our system presents an 34.9% improvement over these standard word prediction systems.

Table 1: Summary of Results

	% improvement over no text prediction	average key presses saved per word predicted	% words predicted
prediction using British National Corpus	17.7	5.4	51.3
prediction using past query corpus (word-based model)	35.7	7.0	75.3
prediction using past query corpus & carrier (word-based model)	35.8	7.0	75.1
prediction using past query corpus & day (word-based model)	35.7	7.0	74.9
prediction using past query corpus & hour (word-based model)	35.9	7.0	75.1
prediction using past query corpus & location (word-based model)	40.1	7.5	78.3
prediction using past query corpus & location (query-based model)	46.4	8.3	80.1

By considering contextual signals in word prediction, we not only decrease the number of key presses needed to enter a query, but we also increase the coverage of words predicted. Whereas the BNC predicts only 51.3% of words, we increase the coverage by to 80.1% when we use the PQC and location signals in the query-based prediction model.

All of our results are based on the assumption that we can only show one suggestion to the user. By imposing this constraint on the user interface, we may not be exploiting the full information gain inherent to the user's context. Next, we discuss variations on the interface, and how they impact the improvement of the system.

6. EFFECTS OF DIFFERENT USER INTERFACES

In addition to exploring various contextual signals, we explored the impact of different user interfaces to the query prediction system. The system presented in this paper assumed one suggestion was shown to the user, and the user was inputting the query from a 9-key keypad. This section will show the effect on the improvements reported when those assumptions do not hold.

6.1 N-word suggestions

Above, we described an interface which will display one query or word suggestion to a user. This is quite likely the simplest interface for the user to understand and use. Here, we explore how much of the potential benefit of our system is lost due to the interface constraint of showing only one suggestion per prefix.

To do this, we measure if we have improved the *probability* of predicting a correct word. At each prefix length we measure the average number of words with probability of occurrence greater than or equal to the probability of the target. This number is the number of words that would be required to be displayed in order for the user to select the correct target word - this statistic reveals by what percent we have decreased the space of word prediction possibilities. The results are shown in Table 2.

By adding the location context, we significantly reduce the number of queries that we would need to show on average to produce the correct word completion. The decrease in words required to be shown is on average 29% better than PQC model.

Table 2: Average number of words with probability of occurrence greater than or equal to the desired word.

Prefix length	PQC Model	PQC+ location Model
1	241.5	121.2
2	39.9	20.7
3	6.7	4.0
4	2.2	1.7

We are not suggesting hundreds of words be displayed to the user; of course this would lead to a terrible user experience. We quantitatively illustrate the reduction in the prediction space only to give a sense of magnitude of information that is lost from the contextual signal when we restrict the interface to one key press.

Additionally, this reduction is significant because it suggests UIs that display more than one suggestion may be more appropriate for query completion. Figure 3 shows the improvement curve as the number of suggestions presented to the user increases. As we can see, after four suggestions, the rate of improvement decreases. This can further guide interface design by providing an upper bound on the number of suggestions which will make a significant impact on the user experience.

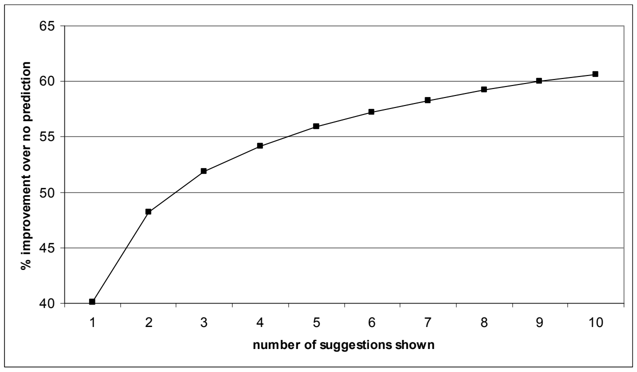

Figure 3: Improvement in the query prediction system as a function of the number of word suggestions shown on the interface.

6.2 QWERTY keyboard

In computing the key presses saved in the experiments outlined above, we assumed query entry occurred with a 9-key keypad, which is prevalent on mobile phones. The impact of using a QWERTY keyboard should not influence the relative improvement context provides for query prediction because the distribution of letters saved is equal to the distribution of letters pressed.

We measured the improvement of the location-based prediction system using the word-based model, assuming the queries were being entered from a full QWERTY keyboard. The improvement over having no prediction was 32.8%. This improvement is slightly deflated from the one reported for query entry on a 9-key keypad because the number of key presses need to "select" a word remained constant at 1 across both keyboards. For example, using multi-tap based measurements, we gained a key press improvement when the user selected a completion when there was only one letter left to type in the query (since the average number of keypresses to specify a letter is 2.2). In contrast, if we assumed use of a QWERTY keyboard, no improvement would have been gained by predicting the last letter of a word.

7. CONCLUSION AND FUTURE WORK

We were able to improve query entry by 46.4% when taking the user's location and the application context into account. This paper presents conclusive evidence that context can improve query entry. We are currently working on designing, implementing and evaluating the usability of prediction interfaces that are integrated with a mobile client application.

There are several challenges to implementing this system on a mobile phone: namely the constrained storage and computation power of mobile devices. In our initial prototypes, in order to resolve problems with the phone's space constraints, we only store parts of the dictionary that are most relevant to the user's current context. Our approach will ease the phone's burden of computation by pre-computing word probabilities on the server side. When there is a change in the user's context, the server will be notified via an http request to the server. The server's response will indicate to the application how to adapt the word dictionary to the user's current context.

Additionally, progress can be made in the design of the user interface. We have shown the significant improvement that can be gained by showing more than one word completion at a time. However, there may be a degradation in user-perceived improvement if more than one completion is shown. Although the number of key presses to enter a query decreases, it make take more time to find and select the desired query completion as the interface will be more cluttered. Designing interfaces that offer more suggestions which can be accessed with a low number of key presses and low cognitive overhead is an interesting avenue for future study. Perhaps the easiest way to quantify the tradeoff of more suggestions and cognitive load is by measuring the time spent entering a query in addition to the key press savings.

Studies are needed to determine whether a non-stable query prediction system is confusing for users. Is it confusing if the letter "s" triggers the completion "sushi" when the user in San Francisco, but triggers the word "soup" when that user visits North Dakota? Should we provide a means for the user to "turn off" context-based suggestions, or to artificially set her context (e.g. by specifying a location other than her current location) ?

In addition to the design, implementation and evaluation of the user interface, there are several immediate avenues for further improving query prediction model itself. First, we can improve upon the model used to generate the predictions. Location based models may be improved by alternate smoothing methods – such as ones that use geographical constraints. In these models, the queries in a nearby city will be taken into account in the query prediction.

We can also incorporate more signals into the query prediction model. For example, with a user's explicit permission, it would be interesting to study the impact of personalized cues – such as an individual's past query behavior, and location history (does the user live in the location? Is she new to it? What queries does she frequently submit in new locations?).

Finally, using context to improve other pre-query tasks should be considered. For example, the task of formulating the query can be improved by suggesting queries to the user which are most relevant to her context before she begins typing. This may create a search interface that eliminates typing altogether from the query entry process.

8. ACKNOWLEDGEMENTS

We would like to gratefully acknowledge the help of Mehran Sahami and Daryl Pregibon for their expertise in statistics, and to Sanjay Mavinkurve for lending his design skills.

9. REFERENCES

[1] Kamvar, M, Baluja S (2006) "A Large Scale Study of Wireless Search Patterns: Google Mobile Search", CHI , pp 701 - 709

[2] Jose, J. & van Rijsbergen, C.J. (2004) "Workshop on Information Retrieval in Context: Report", SIGIR IRiX Workshop

[3] Freund, L., Toms, E.G. (2005) "Contextual search: from information behaviour to information retrieval", Annual Conf. of the Canadian Association for Information Science.

[4] Belkin, N.J., Muresan, G., Zhang, X.M. (2004) "Using User's Context for IR Personalization", SIGIR IRiX Workshop.

[5] Lawrence, S. (2000) "Context in Web Search", IEEE Data Engineering Bulletin, V. 23:3, pp 25-32.

[6] Finkelstein, L. et al. (2001) "Placing Search in Context: The Concept Revisited", WWW, pp 406 - 414

[7] Toms, E.G., Marche, S., O'Brien, H. Toze, S., Trifts,V., Dawe, E. (2004) "Situational Impact on Search", SIGIR IRiX Workshop

[8] Brown, PJ, and Jones G J F (2001) Context-aware retrieval: Exploring a New Environment for Information Retrieval and Information Filtering in Personal and Ubiquitous Computing, V.5 pp 253-263.

[9] MacKenzie, I S, et al (2001) LetterWise: prefix-based disambiguation for mobile text input, UIST, pp 111-120

[10] British National Corpus http://www.natcorp.ox.ac.uk