LIVE SEARCH FOR MOBILE: WEB SERVICES BY VOICE ON THE CELLPHONE

A. Acero, N. Bernstein, R. Chambers, Y.C. Ju, X. Li, J. Odell, P. Nguyen, O. Scholz, G. Zweig

Microsoft Corporation
One Microsoft Way, Redmond, WA 98052

ABSTRACT

Live Search for Mobile is a cellphone application that allows users to interact with web-based information portals. Currently the implementation is focused on information related to local businesses: their phone numbers and addresses, directions, reviews, maps of the surrounding area, and traffic. This paper describes a speech-recognition interface which was recently developed for the application, which allows the users to interact by voice. The paper presents the overall architecture, the user interface, the design and implementation of the speech recognition grammars, and initial performance results indicating that for sentence level utterance recognition we achieve 60 to 65% of human capability.

Index Terms— multimodal, speech recognition, cellphone, user interface, grammars

1. INTRODUCTION

While speech-enabled user interfaces to mobile devices have been heavily researched for several years [1, 2, 3, 4], they are just beginning to reach the market in commercial applications for cellphones that can be freely downloaded [5, 6] or purchased [7, 8]. This first generation of applications is focused on relatively constrained information-access tasks such as finding business phone numbers and addresses, and it combines elements of classic voice-only directory assistance and related applications [9, 10, 11, 12, 13] with a new ability to receive and display information in a multimodal fashion. This paper describes one such recently developed application, *Live Search for Mobile* (LS4M) [14], which has been developed for users of cellphones running *Windows Mobile*.

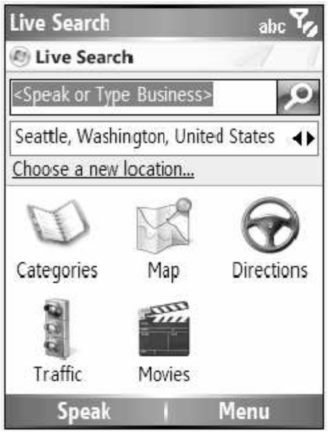
The multimodal and handheld nature of cellphone communication creates numerous challenges and opportunities for speech-enabled search applications. Primary among these is the design of the user interface: users are not initially aware of the capabilities and limitations of speech recognition, and an intuitive interface must be provided for delivering n-best information and soliciting feedback. Even in a constrained application, users may submit complex queries, for example by combining both a request and a location in a single utterance as in "Highlands Restaurant on 10th Street in Bellevue," thus requiring a procedure to tease apart the different pieces of information. Since purely spoken interactions might hide distinctions that are necessary to maintain in written communication, there are further problems related to the visual display of information. For example, in a purely spoken system, homonyms can be collapsed into a single representative as long as this is done consistently in the vocabulary and search indexes. For visual display on a cellphone, however, it is important to maintain the distinction between homonyms. Address normalization is another case where visual representation becomes important: a phrase like "Twenty One Twenty Two Lakeshore Avenue" is much more concisely rendered as "2122 Lakeshore Ave.," so a procedure for rendering numbers must be developed. The overall architectural design of a mobile client poses further challenges: what processing should be done on the phone itself and what should be done server-side? What is the minimal set of national and location-specific grammars and language models that should be designed so that a multipass recognition strategy can share portions of the grammars across passes? The contribution of this paper is to describe solutions to these problems, as implemented in the LS4M application, and present initial performance results.

The remainder of this paper is organized as follows. Section 2 presents an architecture for providing a mobile search capability. In Section 3, we present an effective user interface, and Section 4 describes the design decisions of our speech recognition backend. Performance measures are offered in Section 5, and concluding remarks are presented in Section 6.

2. ARCHITECTURE

The LS4M architecture is organized around a bank of relay servers which mediates between a cluster of machines doing speech recognition, and the MSN Local Search servers. Interactions are initiated by the client application, which depending on its state will either send the relay servers a request for the execution of a specified web service or for the transcription of an utterance. The only computation performed client side is minimal signal processing prior to utterance transmission, and control-flow logic. Client requests are received by a load balancer which forwards the message to an available relay server. At this point, the message is either forwarded to MSN local search, or to a secondary load balancer governing a bank of speech recognition servers. If speech recognition is requested, the utterance is forwarded to an unloaded speech server for processing.

1-4244-1484-9/08/$25.00 ©2008 IEEE



<Speak or Type Business>

Seattle, Washington, United States

Fig. 1. LS4M main screen

This architecture has two notable features. First, the speech servers are completely homogeneous in the sense that every server has access to all grammars and loads the necessary ones on demand. This is in contrast to an approach based on localization where specific servers are dedicated to serving specific locations, and complex load balancing is necessary to ensure that the proper number of servers is dedicated to each location given the time-of-day. Second, the system minimizes latency by engaging in speculative search execution. Once recognition of a business is performed, the entries in the n-best list are entered as queries to MSN Local, and the results cached while the user selects an option.

3. USER INTERFACE

The user interface of LS4M has been designed with the following principles:

1. Keep it simple. Avoid cluttered screens and large numbers of options.
2. Keep the user in control. To avoid unanticipated consequences due to speech recognition errors, seek confirmation before taking an action based on the input.
3. Accomodate the user. Recognition should work even when a user enters a location along with a business into a business-only field.
4. Maintain consistency with text-based search.

Figure 1 illustrates the LS4M main screen. The top two boxes indicate the business name and location that were last accessed; if no business has been accessed in the current session, the “Speak or Type Business” message is displayed. The icons shown at the bottom of the screen show non-search functionality that can be accessed.

If the cursor is moved into the topmost box (as illustrated), the user can click to begin speaking a business name, for example “Overlake Medical Center in Bellevue Washington.” In the case that a location component is present, it is identified as described in Section 4.1, verified with the user, and recognition is re-run with a location specific grammar. The n-best options are presented to the user, and once one is selected, a list of matching businesses is presented by MSN local search (Figure 2).

4. SPEECH RECOGNITION INFRASTRUCTURE

4.1. Grammar Structure

There are three main items the application listens for, and these are reflected in the grammars.

1. An entity - either a business such as “Starbucks”, or an address such as “1 Microsoft Way”
2. A U.S. location, potentially including city, state and zip, such as Redmond Washington
3. An entity together with a location (for example Starbucks near Redmond Washington)

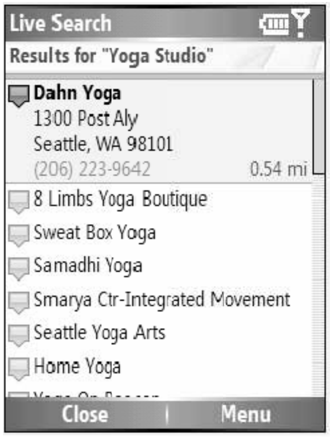
These grammars are implemented as bigram language models with no count cutoffs for business names and addresses, and as a single highly constrained grammar for location recognition, such that only valid combinations of city, state and zip code can be decoded. Business and address grammars exist at both the local and national levels, with a local grammar for each of six hundred geographical areas. These individual components are connected together as necessary to recognize various combinations.

With focus on the main screen as in Fig. 1, the system can recognize either national or location-specific entities. Specifically, it listens for:

- An entity using the grammars tailored to the currently set location, and
- An optional entity plus location using the national grammars.

When the system recognizes either an entity name or a location by itself, it implements a single stage of recognition and confirmation. The recognition result (and any alternate phrases) are displayed in the form of a simple list from which the user can accept either the top result (the default action), tap down to an alternate and accept that, respeak or cancel. The application then displays the search results (for an entity) or updates the current location (for a location). Search results are speculatively streamed to the client after recognition to reduce overall latency, and to further reduce latency, confirmation is skipped with high confidence recognitions. The confidence threshold is chosen to keep the rate of false acceptance of errors below 5%, which results in confirmation being skipped between 25% (for noisy or difficult examples) and 50% (for clearer and easier examples) of the time.

When both a location and an entity are recognized, these are separately confirmed and two stages of recognition are



Live Search

Results for "Yoga Studio"

Dahn Yoga
1300 Post Aly
Seattle, WA 98101
(206) 223-9642 0.54 mi

8 Limbs Yoga Boutique

Sweat Box Yoga

Samadhi Yoga

Smarya Ctr-Integrated Movement

Seattle Yoga Arts

Home Yoga

Fig. 2. Search results

performed. In the first stage the national entity grammar is used with the location grammar and the first stage response consists of the alternative locations recognized, together with entities recognized using the national grammar. The second stage recognition uses a tailored location specific grammar to re-recognize the same utterance. Speculative prefetch of the search results happens for each of the entity recognitions.

4.2. Text Processing

Prior to building the grammars, a significant amount of text processing must be performed to normalize the data, and this is handled differently for the business and address grammars.

4.2.1. Business Name Normalization

The text processing used for business names is a combination of hand-coded rules, along with a novel transduction process used to normalize externally transcribed data. The data used to build the business name grammars comes from three sources: commercial data feeds including Amacai and InfoUSA; transcribed calls courtesy of Tellme; and speech recognition transcripts of successfully automated calls in an existing directory assistance application. This data was processed in three steps: First, each database listing was parsed and a sequence of transformations and replacements applied to produce realistic ways that it might be spoken. For example, the entry “Seaview Chevrolet Pontiac” would generate forms including “Seaview Chevrolet,” “Seaview Chevy Dealer,” and “Pontiac Dealer.” Second, a set of word breaking rules was applied to break words like “airline” into “air line.”

The final step is transduction based normalization. This is intended to correct homonym-related mistakes that occur in manual transcription, and homonym mappings that were performed in certain data sources. Some of these have the property that they can only be corrected using contextual information. For example, “Rite Aid Drug Store” might occur somewhere as “Right Aid Drug Store,” but it would be inappropriate to always turn “Right” into “Rite.” To create the necessary transducer, we compose a lexical transducer with a language model. The lexical transducer maps from words to words, allowing any homonym of an input word to be output. For example, reading “Right” could result in either “Right” or “Rite” with equal probability. The language model is built on formal listing text, so it would see, for example, “Rite Aid Drug Stores,” but never “Right Aid Drug Stores.” The composition of these results in a transducer that takes words on the input, and outputs the homonym-specific word sequence with the highest language model probability.

4.2.2. Address Normalization

The address normalization process consists of a sequence of context-free word replacements, followed by the generation of alternate spoken forms of numbers. The word replacements simply expand abbreviations like “Ave.,” “Trl.,” and “Hwy.” The number normalization is trickier (consider e.g. “2122 Lakeshore” or “31257 Main Street”) and implemented as a recursive procedure. As the base case, the spoken forms of all one and two digit numbers are enumerated. Then rules are defined for interpreting longer digit strings. For example, a three digit string is realized as a one-digit string followed by a two digit string. A four digit string is realized as two two-digit strings, and so forth. A sequence of single digits is always allowed. Thus, the entry “2122 Lakeshore” would be expanded into “Twenty One Twenty Two Lakeshore” and “Two One Two Two Lakeshore.” The generated forms are entered into the language model training text.

4.3. The Recognizer

The speech recognition server that forms the backend for the Live Search service is based on the same recognition technology that ships in Windows Vista and Microsoft Office Communication Server 2007: Speech Server Edition [15]. The recognizer is based on a dynamic network decoder using continuous density hidden Markov models with cross word, context dependent tied state tri-phones. The front-end produces Mel-frequency cepstral coefficients and incorporates dynamic real-time cepstral mean normalization and noise compensation. Currently the training data does not include any live search data and instead comes from a variety of telephony sources, both publicly available corpora as well as internally collected data.

4.4. Address Rendering

Addresses are often long and complex, and concicely rendering after recognition them is important from a user interface standpoint. The primary challenge involves rendering the spoken form of street numbers - which are output from the recognizer as words. For example, “Twenty One Twenty Two Lakeshore Avenue” for “2122 Lakeshore Avenue.”

	1-SA	n-SA	WER	WER-UNK
agree	50.4%	55.6%	43.1%	43.1
disagree	13.2	18.2	64.1	43.3
overall	38.2	43.3	51.2	43.2
interannotator	66.7	-	18.2	13.6

Table 1. 1-best Sentence accuracy (SA), n-best SA, word error rate (WER), and WER less unknown words, for data subsets defined by whether the annotators agreed or not.

To implement address normalization, we experimented with two procedures. Our initial approach was based on a parser with a hand-coded set of rules, and more recently we implemented an alternate procedure based on transduction. In this approach, we create a "lexicon" in which each number in the listing database is treated as a word with multiple "pronunciations" consisting of the possible spoken forms. For example, "2122" is listed as a word with pronunciations *Twenty One Twenty Two*, *Two One Two Two*, etc. These pronunciations are generated via the recursive procedure mentioned earlier. The lexicon (with spoken words on the input and written numerical forms on the output) is then composed with a language model trained on the written database listings. The result is a transducer with spoken words on the input side and rendered forms on the output. To test address normalization, we asked a half-dozen naive researchers to specify the way they would ask for a random sampling of addresses. This resulted in a test set of 300 addresses, on which the deployed rendering procedure achieves a 4% sentence error rate.

5. PERFORMANCE MEASURES

At the time of writing, LS4M is in the initial stages of internal deployment, and to analyze the error rates being attained, about 350 interactions comprising just over 1000 words were transcribed by two of the authors. Each interaction was transcribed twice in order to get a baseline inter-annotator error rate against which to compare the speech recognition error rate. With some frequency, unknown words were present, for example when an obscure town was requested, and these events were annotated with <UNK>. The data was found to be extremely noisy, deriving from two basic reasons:

- Due to the multimodal nature of the application, the cellphone is typically held at arms length, resulting in a loss of over 20dB in SNR.
- Users are prone to use the application in inherently noisy environments, for example, in cars, on the street, and with friends talking in the background

Sentence accuracy and word error rates are shown in Table 1; the data is broken into subsets depending on whether the human transcribers agreed. The rightmost column shows word error rate, excluding errors involving <UNK>.

These results indicate the difficulty of the task: in only about two thirds of the cases did the human listeners agree on the entire utterance. Against this backdrop, the recognizer correctly recognized 38.2% of all utterances at the sentence level - about 57% of human ability. The human word error rate less unknowns is 13.6%, with the recognizer attaining around 43%. If we consider n-best sentence accuracy, with up to nine alternates displayed, then the overall accuracy level increases to 43.3%, or 65% of human capability.

6. CONCLUSION

This paper has described the design and implementation of Live Search for Mobile, one of the first commercially available multimodal voice search applications. LS4M endows *Windows Mobile* cellphones with a voice-enabled data input functionality, and integrates it with search functionality to provide local-business related information. Perhaps the key challenge is the loss of signal-to-noise ratio caused by the fact that users often speak at arms length while looking at the screen. Our system achieves 60 to 65% of human performance under these circumstances.

Acknowledgements

We wish to thank Kyle Oppenheim, Bruce Buntschuh and Shawn Chang for their invaluable advice and assistance.

7. REFERENCES

[1] X. Huang and et al., "MiPad: A next generation PDA prototype," in *ICSLP*, 2000.

[2] J. Lai, "Facilitating mobile communication with multimodal access to email messages on a cell phone," in *Proc. Computer Human Interaction*, 2004.

[3] S. Oviatt, "Taming recognition errors with a multimodal interface," *Communications of the ACM*, vol. 43, no. 9, 2000.

[4] M. Cohen, "Emerging and exotic suditory interfaces," in *Proceedings, 114th Convention of the Audio Engineering Society*, 2003.

[5] "http://www.tellme.com/products/tellmebymobile," .

[6] "http://www.vlingomobile.com/downloads.html," .

[7] "http://www.nuance.com/mobilesearch," .

[8] "http://www.voicesignal.com/solutions/vsearch.php," .

[9] B. Buntschuh, C. Kamm, G. Di Fabbrizio, A. Abella, M. Mohri, S. Narayanan, I. Zeljkovic, R.D. Shard, J. Wright, S. Marcus, J. Shaffer, R. Duncan, and J.G. Wilpon, "VPQ: A spoken language interface to large scale directory information," in *ICSLP*, 1998.

[10] L. Boves, D. Jouvert, J. Sienel, R. de Mori, F. Bechet, L. Fissore, and P. Laface, "ASR for automatic directory assistance: the SMADA project," in *ASRU*, 2000.

[11] C.A. Kamm, K.M. Yang, C.R. Shamieh, and S. Singhal, "Speech recognition issues for directory assistance applications," in *2nd IEEE Workshop on Interactive Voice Technology for Telecommunications Applications*, 1994.

[12] E.E. Jan, B. Maison, L. Mangu, and G. Zweig, "Automatic construction of unique signatures and confusable sets for natural language directory assistance applications," in *Eurospeech*, 2003.

[13] G. Chung, S. Seneff, C. Wang, and L. Hetherington, "A dynamic vocabulary spoken dialog interface," in *ICSLP*, 2004.

[14] "Available by time of publication," .

[15] J. Odell and K. Mukerjee, "Architecture, user interface, and enabling technology in windows vistas speech systems," *IEEE Trans. Computers*, vol. 56, no. 9, 2007.