Rethinking Speech Recognition on Mobile Devices

Anuj Kumar[1], Anuj Tewari[2], Seth Horrigan[2], Matthew Kam[1], Florian Metze[3] and John Canny[2]
[1]Human-Computer Interaction Institute, Carnegie Mellon University, Pittsburgh, USA
[2]Computer Science Division and Berkeley Institute of Design, University of California, Berkeley, USA
[3]Language Technologies Institute, Carnegie Mellon University, Pittsburgh, USA
{anujk1, mattkam, fmetze}@cs.cmu.edu

ABSTRACT
In this paper, we describe our experiences and thoughts on building speech applications on mobile devices for developing countries. We describe three models of use for automatic speech recognition (ASR) systems on mobile devices that are currently used – embedded speech recognition, speech recognition in the cloud, and distributed speech recognition; evaluate their advantages and disadvantages; and finally propose a fourth model of use that we call Shared Speech Recognition with User-Based Adaptation. This proposed model exploits the advantages in all the three current models, while mitigating the challenges that make any of the current models less feasible, such as unreliable cellular connections or low processing power on mobile devices, which are typical needs of speech application in developing regions. We also propose open questions for future research to further evaluate our proposed model of use. Finally, we demonstrate the performance of two mobile speech recognizers that are either used in a lab setting to compare the recognition accuracy against a desktop, or used in real-world speech applications for mobile devices in the developing world.

Author Keywords
Mobile Devices, Speech Recognition, Developing Regions.

ACM Classification Keywords
H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

INTRODUCTION
Speech is the easiest and most common way for people to communicate. Speech is also faster than typing on a keypad and more expressive than clicking on a menu item. For these reasons, speech applications are of particular importance, especially for users with low literacy or little script knowledge, such as those in the developing regions.

On the other hand, mobile internet devices, ultra mobile PCs, internet tablets, smartphones and cellphones have widely proliferated the market, both in developing nations and in the low socio-economic communities of the developed world, sometimes to the extent that users are more likely to have these mobile devices than a personal computer. For instance, according to Gartner, over 139 million smartphones were sold in 2008, an increase of 13.9% over the previous year [1].

IUI4DR, February 13, 2011, California, USA

Accordingly there's a growing interest in developing applications that employ automatic speech recognition (ASR) on mobile devices. However, direct reproduction of algorithms that are suitable for desktop applications is either not possible or often leads to low performance on mobile devices [12]. Since these mobile devices are often used when the person is "on the move", variable acoustic environments and limited resources on the mobile device necessitates special arrangements [11].

Prior research on speech applications in developing regions, such as those for information access [8, 13] or those information management [9] have explored the use of ASR under three different models: embedded speech recognition, speech recognition in the cloud, and distributed speech recognition. The above approaches are limited in either the requirement to have a mobile device with high processing power, or to be used at places with reliable and continuous cellular connection. In this paper, we propose a fourth model of ASR use that we call Shared speech recognition with user-based adaptation, which exploits the advantages of the above three models while mitigating the challenges that they face. This model uses a "decode locally, supervise remotely, then adapt" approach that does not rely on a cellular connection, but instead gains from an intermittent connection whenever available with a central server. It is able to perform robust speech recognition locally, but postpones computationally intensive tasks like user-based adaptation to the server. Such an approach also benefits from user-based adaptation techniques that can account for the large variabilities in use of mobile devices such as noisy environments, small rooms with reverberation of multiple speakers, non-native speech, varying speaker gender, etc.

The contribution, therefore, of this paper is to propose a model of use of mobile ASR systems that can be used at places with unreliable cellular connection. It also proposes open research questions to improve and investigate the type of decision-making algorithms that the proposed model should employ. The rest of the paper is organized as follows: the challenges of ASR systems on mobile devices; description, advantages and disadvantages of ASR systems on mobile devices using three modes: embedded speech recognition, speech recognition in the cloud, and distributed speech recognition; a fourth, proposed model of use and

research questions. Finally, we also present two cases of speech recognition on mobile devices, and sample measurements asserting the practicality of speech recognition on mobile devices, both in the lab and in the real scenarios such as speech-enabled language learning applications in rural parts of India.

CHALLENGES WITH MOBILE ASR

When compared to ASR systems on desktops or servers, implementation of accurate speech recognizers on mobile devices is challenged by several factors, including: limited available storage space (language and acoustic models have to be smaller, which leads to low performance), cheap and variable microphones, often far away from speaker's mouth, no hardware support for floating point arithmetic, low processor clock-frequency (algorithms implement a tradeoff between speed and accuracy), small cache of 8-32 KB, and highly variable and challenging acoustic environments ranging from heavy background traffic noises to a small room with reverberation of multiple speakers speaking simultaneously. Moreover, cellphones consume a lot of energy during algorithm execution [12], which is a significant factor for consideration for speech application in the developing world, where prior deployments have demonstrated that electricity issues could hamper use of mobile applications in everyday settings [5].

MODE 1: EMBEDDED MOBILE SPEECH RECOGNITION

In the case of embedded mobile speech recognition, the entire process of speech recognition, including feature extraction and search happens locally on the mobile device, as shown in Figure 1. However, it is important to note that even though feature extraction is a computationally intensive operation, it consumes only 2% of the processing time in case of medium-sized vocabularies and even less for large vocabulary recognition tasks [12].

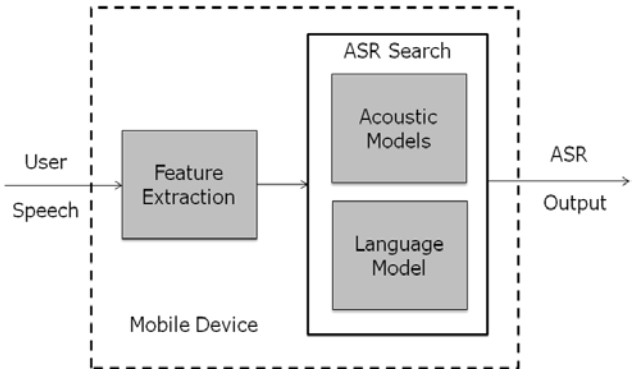

Figure 1: System architecture when both the feature extraction and the ASR search are implemented on the mobile device alone [12]

Advantages

The main advantage of this mode is that it does not rely on any communication with a central server, and hence can work in conditions where there are no networks. Thus, the ASR system is always ready for use, and since the audio data is recorded and decoded on the same device, the recognition is not affected by the quality of the data transmitted. Moreover, there are no additional costs involved with transmitting and receiving information back from the server. Most importantly, such systems are not affected by the latency that's involved with transmitting and receiving information from a server. Latency becomes a major issue in developing regions because cellular data connections in such regions are costly, slow and unreliable with frequent call drops.

Disadvantages

The disadvantage of this approach is that mobile devices are still not comparable to full-fledged servers that can perform complex computations. They lack in terms of speed and memory, which in turn restricts the type of applications which can be supported [6]. To achieve reliable performance, modifications need to be made to every sub-system of the ASR to take both factors into account.

MODE 2: SPEECH RECOGNITION IN THE CLOUD

In the case of speech recognition in the cloud, both the feature extraction and ASR search are shifted to the server with mobile device in constant communication with the server, as shown in Figure 2.

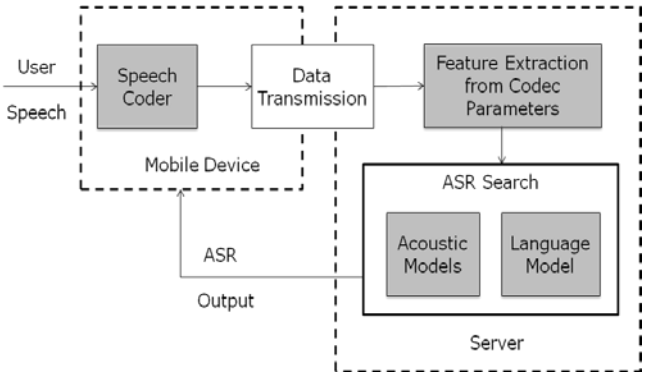

Figure 2: System architecture when both the feature extraction and the ASR search are implemented on the server end with mobile devices encoding the speech signal before transmitting the encoded bits to the server [12]

Advantages

Speech recognition in the cloud, often referred to as network speech recognition, is a commonly used mode, and has several advantages to it: the burden of post-processing audio and extracting meaning from it is given to a powerful dedicated machine. This machine is generally a high configuration server capable of doing complex computation fast, which is essential for real-time speech recognition systems. This offers significant advantage in terms of speed and accuracy. In addition, the implementation of ASR may be confidential to the developers, and this approach avoids any local distribution or installation, which provides an easy way to upgrade or modify the central speech recognition system.

Another significant advantage of this mode is that it can be used for speech recognition with low-end mobile devices such as cheap cellphones. This can be particularly useful for users in the developing region who own low-end cellphones that are only capable of making and receiving voice calls, and may not support complex speech recognition tasks.

Disadvantages

In spite of the advantage of increased resources, this mode of recognition has some disadvantages to it that makes this mode less attractive. One of the most important cons of this approach is the performance degradation caused by low bit-rate codecs, which becomes severe in the presence of data transmission errors or background noise [12].

Another disadvantage of this mode is that acoustic models on the central server need to account for large variations in the different channels (quality of microphone and audio card) on each terminal mobile device, which spans a large number of conditions. Furthermore, an important issue related to ASR design in this mode is the server-side architecture which should be capable of supporting requesting from hundreds of clients simultaneously and processing them without any additional queuing delays.

Another reason that makes this option less attractive is that each data transfer over the telephone network can cost money for the end user. This is a significant problem for users in the rural parts of developing regions who would want to use applications that provide a continual service with no or just a one-time fee. Server-based speech dialog systems, as commonly used in customer self care, are a sub-class of this mode.

MODE 3: DISTRIBUTED SPEECH RECOGNITION

In this mode, the speech recognition load is divided between the server and the end mobile device. The computationally less costly part of ASR i.e. feature extraction is done locally on the mobile device, while the search for the most likely recognition hypothesis is done on the server.

Advantages

The advantages of this mode are similar to those of the previous mode where the entire speech recognition is done at the server end. In addition, since ASR does not really need high quality speech, but rather a set of acoustic parameters, such as feature vectors, this mode can benefit from no loss in data due to speech coding, transmission and decoding at low bit-rates, as in the previous mode. Subsequently, better methods to improve word error rates (or similar metrics depending on the requirements of the specific task) can be developed, since normalization and adaptation can be specific to device and speech codec.

Disadvantages

The major disadvantage of this mode still remains cost and the need of continuous and reliable cellular connection, as in mode 2, although the requirements are somewhat relaxed. Also, since the feature extraction is happening on the mobile device and the ASR search takes place on the server end, there's a need for standardized feature extraction processes that account for variabilities arising due to differences in channel (microphone and audio data card), multi-linguality, variable accents, Lombard effect [4], and gender differences, etc.

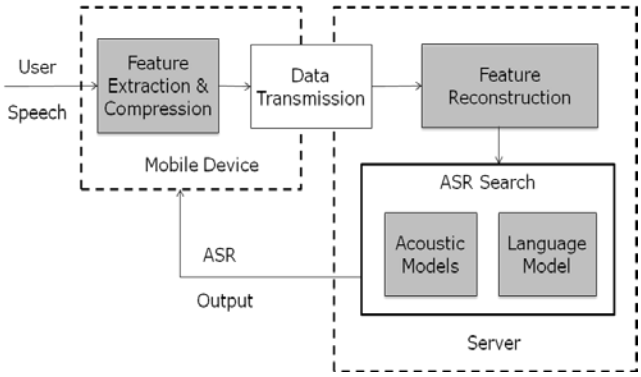

Figure 3: System architecture when the speech recognition is split across the mobile device and the server. Feature extraction is done on the mobile device, and the ASR search is implemented on the server [12]

MODE 4: SHARED SPEECH RECOGNITION WITH USER BASED ADAPTATION (PROPOSED MODEL OF USE)

This mode aims to exploit the major advantages of all the above three modes, while also mitigating the disadvantages of each of them. In this mode, a continuous cellular connection is not required for the speech application to function. All the major sub-systems of an ASR are located on the mobile device, and the device can perform speech recognition tasks even under no network conditions, as shown in Figure 4. However, whenever there's an intermittent cellular connection available, the mobile device sends the extracted feature vectors with metadata information of the user and device, such as noise levels, channel information, decoded outputs etc. to the server. This enables the server to evaluate the recognition performance for each user independently with a much larger vocabulary and acoustic database, and then recommend adaptations. These suggested adaptations would often be user context-based, for instance, a user who's using the application in a noisy background is likely to need noise filtering adaptations in the acoustic models stored locally at the mobile device. Similarly, a non-native speaker of a language is more likely to need adaptations to the pronunciation dictionary in the local ASR. Hence, even though the server is not responsible for decoding the speech signal while the application is in use, in this mode, the server's compute power can be helpful to recommend user-based adaptations to each individual mobile device, while using larger, shared resources for correcting the local recognition.

The proposed architecture combines the most important advantages of the two previous server-based approaches

(high recognition accuracy, updates and maintainability, moderate requirements on mobile hardware), with the advantage of local ASR, e.g. the ability to function even without network connectivity. Centralized server power however is used not to perform recognition, but to ensure that each instance of distributed recognition works well for the limited set of conditions it encounters. As each mobile phone tends to be used by a few users and in a few conditions only, this subset can be covered successfully by existing mobile devices, if trained or adapted accordingly. As an additional benefit, server capacity has to be provided only for average, not peak use, because adaptation does not have to be provided in real-time.

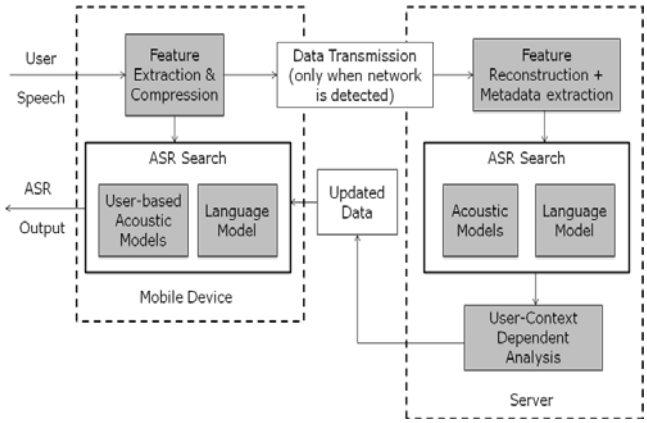

Figure 4: System architecture when the speech recognition is done on the local mobile device, but whenever there's intermittent network connection available, periodic updates for adaptation and improving the recognizer performance on the mobile device are received from the server. These updates are based on each user's acoustic and meta-data.

Future Questions to Explore

The success of this "decode locally, supervise remotely, then adapt" approach depends on the degree, to which the local speech recognizer can be reconfigured (adapted), depending on the server's analysis. We are not aware of published literature or work, which evaluates adaptation of ASR systems not only with respect to performance gain, but also with respect to the complexity of modifications to the original system, and the amount of data which has to be transmitted back to the mobile speech decoder. Possible strategies include:

- Device-specific pre-processing to achieve noise-robustness or signal normalization,
- User-specific vocabularies and language models/ grammars, which model dialectal and idiolectal variations as well as accented speech,
- Speaker-specific acoustic models, or feature- and/ or model-based adaptation strategies,
- Tuning of parameters such as language model weights, end-pointing, etc.

The goal is to use adaptation to eventually achieve speaker-specific recognition performance on a low-resource mobile device, while only general-purpose models and algorithms have to be shipped or installed on the device one-time. Given that there are numerous types of adaptations that are possible in general, the exact and most needed adaptation depends on the condition that the user is in, for instance, noisy backgrounds vs. non-native speech. Research is therefore needed to identify the type of adaptation that will be most beneficial to a user or device, if only a certain amount of data is expected to be transmitted the next time a device connects to the server again. Moreover, this mode of use is also capable to adapt as the user's context changes over time. For instance, if a non-native speaker achieves native speaker-like pronunciation, the server analysis can identify another metric such as noise that affects the speech recognition most in the most recent context, accounting for overall user history as well.

Under this scheme of adaptation, the server is able to independently analyze the acoustic features using a much larger acoustic database as reference, and compute a more accurate recognition hypotheses. These more accurate hypotheses play a dual role: (i) they help in the server-end analysis to compare and contrast against the hypothesis that was generated at the device-end and identify factor(s) that affect recognition performance most in user's context, and (ii) they can also be used as speech labels to adapt the recognizer at the device-end when communicated back to the device.

A requirement, however, of the proposed approach is that speech recognition will be used repeatedly (not just occasionally), and by a low number of speakers, perhaps restricted to the number of members in a typical rural household. Also, while the size of feature that are offloaded to the server for analysis are application specific, it is expected to be equivalent to be about 2-4 minutes worth of speech data per day. A typical "voluntary" usage of mobile phones applications (apart from other uses like voice calls and SMS) in rural India was indicated to be approximately 17 minutes per day [5], and since only a smaller part of a speech application requires the user to speak, we expect the speech data to be about 2-4 minutes.

This architecture is significantly different from the cloud-based or distributed mode of recognition on several levels: First, the speech recognition is always happening on the device end irrespective of the availability of the cellular connection which makes the applications usable at all time and places. The server power is used for analysis for improving the speech recognition performance in the context of the user's environment, as well as other factors such as the network bandwidth or the device processing power available for the user. Second, the adaptations leads to per-client based recognition. Research and results on the latter idea is also beneficial for developed-country contexts, where distributed mode of speech recognition is already in

use. For instance, metadata like Caller ID could be used to maintain different acoustic models for different users.

MOBILE BASED RECOGNIZERS

Following are the two ASR's that were tested on a mobile device for speed and accuracy.

Pocketsphinx

PocketSphinx [2, 10] is a small-footprint, continuous speech recognizer that is an adapted version of Sphinx-II for applications on mobile devices. It offers an easily accessible, open-source, robust mobile speech recognition solution. However, although PocketSphinx is still being improved, it is based on the older Sphinx-II engine which uses semi-continuous acoustic models.

Sphinxtiny

We have adapted CMU's Sphinx3.x open-source large-vocabulary continuous speech recognition system to support use on mobile computing platforms such as Nokia internet tablets or smartphones. We use a fixed-point MFCC front-end, and have accelerated parts of the Gaussian evaluation in assembly. This significantly increases the recognition engine's performance on devices that only support soft floating point calculations. SphinxTiny [14], is designed to merge with new releases of the evolving Sphinx 3.x engine.

BASELINE PERFORMANCES

Results in lab

We evaluated the accuracy and speed of SphinxTiny along with the baseline Sphinx-3.7 system and the open-source PocketSphinx-0.5 system on two different platforms. First, in order to determine the performance of the systems in a resource-rich Linux environment, we used a PC running Ubuntu Linux 8.04 in VMWare Server 1.0.6 on top of Microsoft Windows Server 2003 with an Intel Pentium D clocked at 2.8 GHz with 4 GB of RAM. Second, in order to determine the performance on a resource constrained mobile device, we used a Nokia N800 Internet Tablet running Maemo Linux OS2008 with a TI OMAP 2420 ARM processor clocked at 330MHz with 128 MB of RAM.

To provide both a reference point for comparison with other speech recognition systems, we present accuracy and speed on two disparate corpora: DARPA Resource Management (RM-1) corpus [3], and the ICSI Meeting Recorder (ICSI) corpus [3]. The RM-1 corpus was smaller and had about 1600 training utterances, whereas ICSI corpus was much larger with 90314 training utterances. For RM-1, we used 1000 tied Gaussian Mixture Models (senones) and 30080 context-dependent triphones, whereas for ICSI, we had 1000 senones and 104082 context-dependent triphones. For RM-1, we use a bigram statistical language model with a vocabulary of 993 words and a language weight of 9.5. For ICSI, we use a trigram statistical language model with a vocabulary of 11908 words and a language weight of 9.5. For testing, we select 365 random utterances were selected from the RM-1 corpus, and 400 from the ICSI corpus. The hypotheses were scored using sclite 2.3 from the NIST SCTK Scoring Toolkit version 1.3.

The word error rate (WER) and sentence error rate (SER) listed in Table 1 and 2 represent the percent total errors as reported by the NIST SCTK toolkit when comparing the one-best hypothesis with the human transcription of the utterances. Speed is measured in xRT, which stands for times real time and represents the speed of decoding the chosen utterances in seconds.

Desktop	xRT	WER	SER
PocketSphinx	0.05	6.0%	28.2%
Sphinx-3.7	0.19	7.3%	34.2%
SphinxTiny	0.37	7.3%	35.1%
Mobile			
PocketSphinx	0.53	6.0%	28.2%
Sphinx-3.7	24.34	7.3%	34.2%
SphinxTiny	2.58	7.3%	35.1%

Table 1: Recognition speed and accuracy comparisons using the RM-1 corpus - 365 decoding utterances, 1600 training utterances

As can be seen from Table 1 and 2, on a small vocabulary task i.e. using the RM-1 corpus PocketSphinx outperforms SphinxTiny on both accuracy and speed; however, as the complexity of the acoustic and language models increases, SphinxTiny's accuracy is better than PocketSphinx. From our experimentation, we find that PocketSphinx is superior when using small acoustic and language models for real-time recognition, but for tasks that allow larger delays in exchange for better accuracy, SphinxTiny is a better choice.

Desktop	xRT	WER	SER
PocketSphinx	1.20	55.1%	72.8%
Sphinx-3.7	0.75	39.6%	69.5%
SphinxTiny	1.23	39.4%	70.3%
Mobile			
PocketSphinx	10.65	55.1%	72.8%
Sphinx-3.7	68.49	39.6%	69.5%
SphinxTiny	8.91	39.4%	70.3%

Table 2: Recognition speed and accuracy comparisons using the ICSI corpus - 400 decoding utterances (from English speakers), 90314 training utterances

Results from the field

We evaluated the accuracy of the speech recognizer running locally on the mobile device in a non-lab setting to demonstrate the accuracy of a local ASR and its improvement with context-dependent adaptation. To do so, we implemented two language learning games for Nokia N810 cellphone and deployed them in rural India. Such an application aimed to improve English vocabulary knowledge for non-native speakers of English who had a strong desire to learn it for upward social mobility.

For speech recognition in these games, we used PocketSphinx since it performed better than TinySphinx on small vocabulary tasks. The games aimed at eliciting single words in isolation i.e. isolated word recognition task. In order to train the recognizer for user's voice, we recorded over 5 hours of data from 50 rural children (approximately 6500 utterances), equally divided across gender and grades 4-5. Each utterance in the dataset was labeled with the correct word that it represented at the time of recording itself, and these labeled inputs were used for training the speech recognizer. (We realize that in the above proposed mode 4, such "correct" labels will not be available for adaptation initially at the user end, but a "near-perfect" set of labels can be generated at the server and a selected subset that was incorrectly decoded at the end device be communicated back to the device for adaptation.)

Since this application required recognition of words in isolation, we calculated the accuracy of speech recognition only in terms of word error rate (WER). To do this, we built the acoustic model from speech utterances of 20, 30 or 40 speakers and then tested them on utterances of 10 speakers. The language model was a unigram model for isolated word recognition of selected 30 words. Even for this simple model, the WER seemed to improve with the amount of training data and adaptations to the pronunciation dictionary, as shown in the Table 3.

Amount of Training Data (# of Speakers)	WER without Dictionary Adaptation	WER with Dictionary Adaptation
20	44.1%	42.6%
307	42%	40.1%
40	37.9%	35.2%

Table 3: Indicates the WER with and without the pronunciation dictionary adaptation on 20, 30 and 40 speakers training dataset, and a test dataset of 10 speakers. Each speaker's dataset comprised of 125 utterances.

CONCLUSION

We have summarized the advantages and disadvantages of three existing ASR systems that are currently used for speech applications on mobile devices and proposed a fourth model of use that mitigates the challenges in the developing regions. Our experiments, both in lab and on the field point towards feasibility and applicability of doing speech recognition locally on mobile devices. However, the support of a server – even under limited cellular connection – to assist in complex user-based adaptations would boost recognition accuracy over time. This paper proposes a few research questions to explore the design of decision-making algorithms that can inform the design of speech applications in developing regions, and encourages more researchers to experiment with the proposed ASR model of use i.e. shared speech recognition with user-based adaptation.

ACKNOWLEDGEMENTS

We thank the reviewers for their valuable comments.

REFERENCES

1. Gartner, 2009. Gartner says worldwide smartphones sales reached its lowest growth rate with 3.7 per cent increase in fourth quarter of 2008. Press release.
2. Huggins-Daines, D., Kumar, M., Chan, A., Black, A., Ravishankar, M., and Rudnicky, A.I. PocketSphinx: a free, real-time continuous speech recognition system for handheld devices. In *Proc. ICASSP-06*, pp. 185–188, Toulouse, May 2006.
3. Janin, A., et al. The ICSI Meeting Corpus. In *Proc. ICASSP-03*, Hong Kong, April 2003.
4. Junqua J.C. The Lombard reflex and its role on human listeners and automatic speech recognizers. *J. Acoust. Soc. Am.* **93** (1): 510–24, 1993.
5. Kumar, A., Tewari, A., Shroff, G., Chittamuru, D., Kam, M., and Canny, J. An exploratory study of unsupervised mobile learning in rural India. In *Proc. of ACM Conference on Human Factors in Computing Systems (CHI '10)*, Atlanta, Georgia, April 10-15, 2010.
6. Novak, M. Towards large vocabulary ASR on embedded platforms. In *Proc. of InterSpeech*, 2004.
7. Pallett, D.S., Fiscus, J.G., and Garofolo, J. DARPA resource management benchmark test results June 1990. In *Proc. Workshop on Speech and Natural Language – Human Language Technology*, Hidden Valley, June 1990.
8. Patel, N., Agarwal, S., Rajput, N., Nanavati, A., Dave, P., and Parikh, T. A Comparative Study of Speech and Dialed Input Voice Interfaces in Rural India. In *Proc. of ACM Conference on Human Factors in Computing Systems* (CHI), 4-9 April 2009, Boston, US
9. Patel, N., Chittamuru, D., Jain, A., Dave, P., Parikh, T. Avaaj Otalo - A Field Study of an Interactive Voice Forum for Small Farmers in Rural India. In *Proc. of ACM Conference on Human Factors in Computing Systems (CHI)*, 2010.
10. PocketSphinx 0.6 release | CMU Sphinx – Speech Recognition Toolkit, http://cmusphinx.sourceforge.net/2010/03/pocketsphinx-0-6-release/ (last accessed October 31, 2010).
11. Rose, R.C., Partharathy, S. A tutorial on ASR for wireless mobile devices. In *International Conference on Spoken Language Processing*, 2002.
12. Schmitt, A., Zaykovskiy, D., Minker, W. Speech recognition for mobile devices. *International Journal of Speech Technology*, vol. 11, number 2, pp. 63-72, 2008.
13. Sherwani J. Speech Interfaces for Information Access by Low Literate Users, Ph.D. Thesis. Carnegie Mellon University, Pittsburgh, USA. 2009.
14. SphinxTiny – Sphinx3.x for Mobile Devices, http://www.cs.berkeley.edu/~eomer/SphinxTiny/SphinxTiny-0.7.html (last accessed October 31, 2010).