# Multimodal Text Entry on Mobile Devices

Bo-June (Paul) Hsu[1], Milind Mahajan[2], Alex Acero[2]

[1]MIT & Microsoft Research

[2]Microsoft Research

## 1 Motivation

Text entry on mobile phones is growing in popularity due to increasing use of applications such as SMS and e-mail. Mobile phones and mobile devices in general do not have a keyboard which is as convenient as that on the desktop computers. Mobile phones in particular tend to have only a numeric keypad on which multiple letters map to the same key. Users currently use methods such as multi-tap (“this” = 8 44 444 7777) or Tegic Communications’ T9 (“this” = 8447) to enter text using a numeric keypad. Novice users of these methods achieve text entry rates of 5-10 words per minute [1].

## 2 Multi-modal combination of speech & keypad

Speech has a high communication bandwidth which is estimated at about 250 words per minute [2]. However, text entry throughput using automatic speech recognition (ASR) is much lower in practice due to the time spent by the user in checking for and correcting the ASR errors which are inevitable with the current state of the art ASR systems. We will demonstrate a system which uses a combination of speech and keypad input to improve the overall text entry experience for the user of the mobile devices.

## 3 Overview of operation

The user presses and holds the microphone button on the side of the device and speaks a sentence. The user is then presented on screen with the best hypothesis and a selection list for only the first word of the sentence. If the best hypothesis word presented on screen is correct the user presses OK button (a designated confirmation key on the keypad). Otherwise, if the desired word is in the alternates list, the user navigates to it using the up-down keys and presses the OK button to confirm it. Alternatively, the user starts entering the word using the keypad. The algorithm re-computes the best hypothesis word and the alternates list using the a-posteriori probability obtained by combining the information from the following sources:

- the keypad entries for the prefix of the word
- speech recognition result lattice
- words before the current word which have already been corrected
- language model

After, the user correctly enters the desired word; the process is similarly repeated for the subsequent words in the sentence.

# 4 Key Features

### - Combination of speech and keypad

ASR is inherently ambiguous and the results in a probability distribution over word sequences. Entering a prefix of a word with a keypad will also not lead to a unique word hypothesis in general. However, since the ambiguities are orthogonal to some extent between the two modalities, the combination helps to improve the overall performance. As an illustrative example, consider that the words “good” and “home” both correspond to the same key sequence 4663# under T9 but are not highly confusable for ASR.

### - Continuous speech with sequential commit

At first glance, the combination of continuous speech ASR with a word by word correction mechanism (sequential commit) may appear to be sub-optimal and is certainly counter-intuitive. However, this is a key innovation which we believe contributes to a better overall user experience given the current state of the art for ASR on mobile devices.

ASR errors often involve segmentation errors. Consider the famous (though hypothetical) example of the phrase “recognize speech” being recognized by ASR as “wreck a nice beach”. Showing the full ASR result leads to difficult choices for the correction interface. Which words should the user select for correction? When the user attempts to correct the word “wreck”, should it cause the rest of the phrase to change? How would the user feel about other words changing as a side-effect of corrections?

We avoid all these issues by presenting word by word alternates sequentially from left to right. In this hypothetical example, the user would perhaps select the word “recognize” as the second alternate to word “wreck” and our algorithm will probably present “speech” as the top hypothesis for the next word given the context of the previous correction.

Word by word text entry is already a familiar user interface for the users of the mobile devices. Providing the same user interface for both speech assisted text entry and keypad only text entry allows the users to switch seamlessly between the two modes which is advantageous since the users may not use speech all the time even if it leads to better throughput due to social considerations.

### - ASR latency hiding

On mobile devices, ASR result latency can be a problem. Our user interface with sequential commit allow us to take advantage of the intermediate ASR hypotheses and start presenting the word hypothesis to the user before the ASR has completely finished.

### - Graceful degradation

We also consider the language model (a-priori word probabilities in context) in addition to ASR lattice in creating the word hypotheses. This allows us to hypothesize words which are not in the ASR lattice and also leads to more graceful degradation in the presence of ASR errors.

References:

[1] C. L. James & K. M. Reischel, “Text input for mobile devices: comparing model prediction to actual performance”, Proceedings of the SIGCHI conference on Human factors in computing systems, 2001.

[2] M. Kolsch & M. Turk, “Keyboards without Keyboards: A Survey of Virtual Keyboards”, University of California at Santa Barbara Technical Report 2002.