Speech Repair: Quick Error Correction Just by Using Selection Operation for Speech Input Interfaces

Jun Ogata and Masataka Goto

National Institute of Advanced Industrial Science and Technology (AIST)
1-1-1, Umezono, Tsukuba, Ibaraki 305-8568, JAPAN
{jun.ogata, m.goto}@aist.go.jp

Abstract

In this paper, we propose a novel speech input interface function, called "*Speech Repair*" in which recognition errors can be easily corrected by selecting candidates. During the speech input, this function displays not only the typical speech-recognition result but also other competitive candidates. Each word in the result is separated by line segments and accompanied by other word candidates. A user who finds a recognition error can simply select the correct word from the candidates for that temporal region. In order to overcome the difficulty of generating appropriate candidates, we adopted a *confusion network* that can condense a huge internal word graph of a large vocabulary continuous speech recognition (LVCSR) system. In our experiments, almost all recognition errors were corrected and the effectiveness of speech repair was confirmed.

1. Introduction

Automatic speech recognition (ASR) technology has made significant progress over the years due to advances in algorithms, architectures, signal processing, and hardware. However, current ASR systems are error prone, especially in real-world environments. Despite research efforts to improve the recognition accuracy, it can be stated that a certain amount of recognition errors will never be removed. This implies that a user interface that allows a user to correct ("*repair*") recognition errors must be designed. So far, several error correction techniques have been proposed and developed. For example, some commercial software such as IBM's ViaVoice [1] provide a correction function that displays an alternative word list for an erroneous word region, which a user finds and specifies by using a mouse or keyboard. Speech-based error correction techniques that allow a user to correct errors by respeaking or spelling out have been presented [2]. As an extension of this approach, a multimodal error recovery system that uses respeaking, spelling out loud, and handwriting has been presented [3]. However, in comparison to traditional keyboard or mouse input, it is awkward to correct errors on speech input interfaces because a user has to undertake the following operations after speaking: find, specify, and correct each error in the recognition results.

This paper describes an error correction interface for speech input called *speech repair* in which recognition errors can be quickly and easily corrected by users. During the speech input, the speech-repair function successively displays not only the usual word sequence as a recognition result but also other competitive candidates that indicate alternative word hypotheses obtained in the decoding process of the speech recognizer. Since candidates for almost all words in the recognition result are always automatically displayed on a screen, the user can correct erroneous words merely by selecting the appropriate candidate even while speaking to the recognizer. The speech-repair function is thus efficient and easy to use: errors can be handled by simply clicking automatically displayed candidates without waiting for the end of an utterance.

In the following sections, we explain the basic concept of speech repair and then describe the design and implementation of a speech input interface with the speech-repair function. We show that experimental results from 25 subjects indicated the effectiveness of speech repair.

2. Speech Repair

In general, current ASR systems display only a word sequence to the user as the recognition result. Consequently, the following operations should be performed for each error when the user wishes to correct recognition errors:

1. Find each erroneous word or phrase in the recognition result (word sequence).
2. Specify the region of the found word or phrase by using some devices such as a mouse, keyboard, or pen.
3. Replace the erroneous word region with the correct word or phrase by respeaking, typing, handwriting, etc.

Speech repair aims to reduce such a burden on the user for error correction by enabling the user to efficiently and simultaneously perform the above mentioned operations.

2.1. Basic correction function

Figure 1 shows an overview of the speech-repair function. During the speech input, our function displays not only a 1-best word sequence but also a numbered list of N-best competitive candidates for each word (not for a sentence). The competitive candidates indicate alternative internal words that a speech recognizer generates as word hypotheses in the decoding process. Each word in the word sequence is separated by vertical line segments and its competitive candidates are listed below it. A user can replace an erroneous word with the appropriate candidate merely by finding and clicking it. The number of competitive candidates for each segment corresponds to the ambiguity of the recognition result for that segment; a segment for which the speech recognizer has assessed the recognition result tends to have many candidates. This correlation helps a user to efficiently find the erroneous segment by looking at the number of competitive candidates. As shown in Figure 1, a blank candidate – called *deletion candidate* – is always included in the list of competitive candidates for each segment. This blank candidate plays an important role – it assists in deleting any recognition result from the segment: by simply clicking this candidate, a user can remove the word in that segment and make it empty. Consequently, the substitution and deletion operations on each word in the recognition results can be seamlessly executed through the same selection (clicking) operation. The

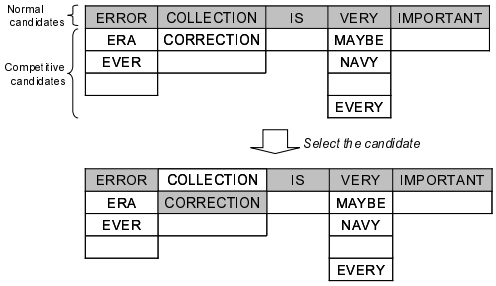

Figure 1. ***Overview of the speech-repair function.***

competitive candidates are sorted in the order of scores (likelihood) calculated by the speech recognizer to enable the user to quickly find the correct candidate by merely seeing from the top of the list.

Although the basic concept of correcting errors using our speech-repair function is simple, current ASR systems, especially LVCSR, cannot directly provide good candidates such as those provided by our function, because N-best sentences of recognition results are not sufficient for candidates displayed on a screen. Since an internal hypothesis network in the decoding process is very huge, the number of candidates in such a network is too high for a user to easily find the correct candidate. In our speech-repair function, we solve this problem by using an efficient intermediate recognition result to generate the competitive candidates so that effective error correction can be achieved even with a LVCSR system.

2.2. Immediate correction function

In order to develop a useful speech interface, it is important to provide immediate and successive feedbacks of the results of the speech recognizer to the user. Although some ASR systems have the function of providing successive feedback of 1-best recognition results to the user, alternative results such as competitive candidates cannot be displayed successively to the user. In this situation, the user cannot begin the error correction operation until the recognition process is completed. Consequently, current speech input techniques have a tendency to be time consuming and laborious in comparison to keyboard-mouse input techniques for text creation tasks [4]. In current speech input techniques, it takes extra time for a user to complete the error correction actions of (1) finding erroneous words and (2) moving a pointer to the location of each erroneous word by using a mouse or keyboard.

From this viewpoint, we introduce an additional function, called *immediate correction function*, in which competitive candidates are successively displayed along with the 1-best recognition result and a user can immediately correct erroneous words by selecting candidates. The immediate correction function enables the user to select a competitive candidate not only when the recognition process is complete but also whenever the user finds erroneous words.

2.3. Intentional suspension function

Although the immediate correction function can reduce the time required for error correction, a user sometimes does not have sufficient time to correct all errors while uttering the subsequent text. Therefore, the user may wish to have time to correct the errors by just suspending an utterance at an arbitrary position, i.e., the user wants the recognizer to temporally stop the decoding process and wait for the error correction to be performed. However, the current ASR systems do not allow a user to stop

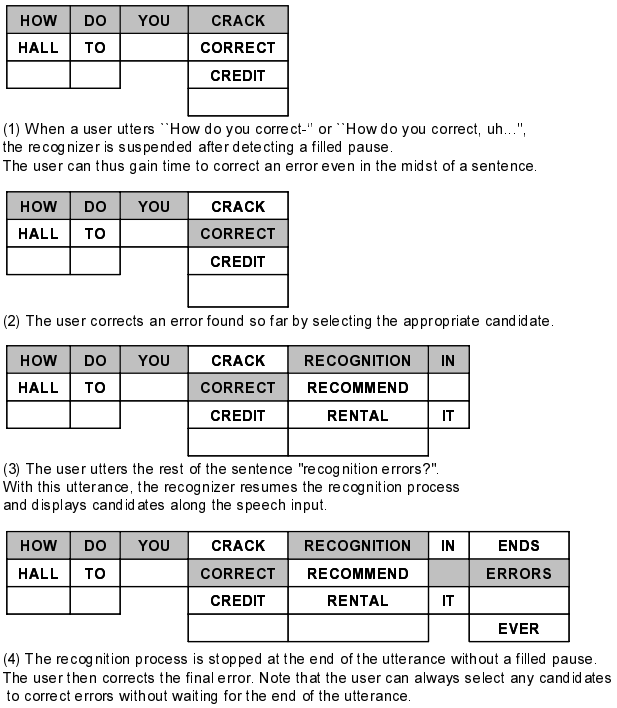

Figure 2. ***Overview of the error correction with intentional suspension function.***

an utterance at an arbitrary position because such a suspension tends to cause incorrect recognition.

Therefore, we introduce an additional function, called *intentional suspension function*, that enables a user to intentionally suspend and resume the recognition process. With this suspension, the user can stop uttering and correct errors found so far. When the user utters the rest of the sentence, the recognizer resumes the recognition process during the speech input. A *filled pause* (the lengthening of a vowel during hesitation), which is a nonverbal component of human speech, is used to trigger this suspension. In actual human-human conversation, the filled pause is a natural hesitation indicating that a user is thinking of or recalling a subsequent word or phrase. Thus, the user can naturally suspend the utterance by uttering a filled pause and can select the correct candidate. Figure 2 shows an overview of the error correction using this intentional suspension function.

3. Implementation of speech repair

In this section, the technical details of developing speech input interfaces with the speech-repair function are presented.

3.1. Competitive candidate generation

In order to develop the speech-repair function, an effective display of the competitive candidates, as shown in Figure 1, is indispensable. The competitive candidates are generated from the intermediate recognition results obtained from the decoding process of the speech recognizer.

Conventional ASR systems generate an N-best sentence list or a word graph (lattice) as an intermediate recognition result. However, the number of candidates in the word graph or the N-best sentence list is too high in the case of LVCSR. Further, a competitive relationship among candidates cannot be explicitly expressed. Accordingly, the user cannot understand and find erroneous words in such conventional intermediate recog-

nition results.

To solve this problem, we adopt the *confusion network* that is a simple network representing the intermediate recognition result and is obtained by condensing a huge word graph. The confusion network was originally introduced in the context of a recent decoding algorithm, *word error minimization*, which minimizes the word error rate of the recognition results rather than the sentence error rate [5]. Our original idea is to apply such an efficient intermediate recognition result to generate competitive candidates for efficient error correction.

In order to obtain the confusion network, the huge word graph is condensed by using the following multiple alignment algorithm:

1. Compute the posterior probability of each word candidate (link) of the word graph.
2. Build equivalence classes, each of them consisting of all the links with the same word-label identical starting and ending times.
3. Merge equivalence classes that contain identical word label by judging their temporal similarity on the basis of overlapped temporal intervals (*intra-word clustering*).
4. Group equivalence classes if they correspond to different words with so-called phonetic similarity (*inter-word clustering*).

Each link of the confusion network has a posterior probability that expresses a possibility of existence in the class and a competitive probability among the other candidates in the class. Since the word candidates of each class are sorted by the value of the posterior probabilities, plausible word candidates in a recognition result are located at the upper positions in the class.

3.2. Method of immediate correction function

In the immediate correction function, the method of displaying the competitive candidates to the user as quickly as possible is important. Therefore, we extended the speech recognizer to generate a confusion network successively at fixed (predefined) regular intervals (500 ms in our current implementation).

At the regular intervals, a partly word graph from the beginning of the utterance to the current frame is generated by backtracking on each survived word hypothesis at the current frame. Subsequently, the confusion network is generated from the partly word graph using the multiple alignment algorithm at the regular intervals. Since typical N-best search methods for generating huge word graphs tend to have large computational cost and are time consuming, we employed a fast search method for LVCSR, called "constrained back-off connection" [7].

3.3. Method of intentional suspension function

The intentional suspension function is based on two methods: filled-pause detection and utterance (endpoint) detection. When a filled pause is detected, the speech recognizer is suspended and recognition results (internal hypotheses and time information) of the current frame are temporally preserved. At this point, the recognizer ignores the interval (frames) when the filled pause is being detected. Subsequently, when the beginning of an utterance is detected (using a typical endpoint-detection method that employs short-time energy), the speech recognizer resumes the recognition process from the preserved internal hypotheses.

To detect filled pauses in real time, we use a robust filled-pause detection method [6]. This is a bottom-up method that can detect a lengthened vowel in any word using a sophisticated signal-processing technique. It determines the beginning and end of each filled pause by finding two acoustical features of filled pauses – small fundamental frequency transitions and small spectral envelope deformations.

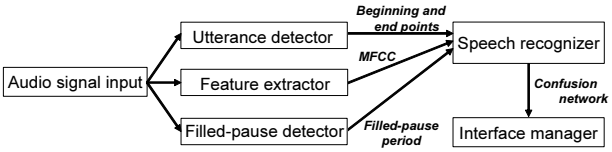

Figure 3. ***System architecture.***

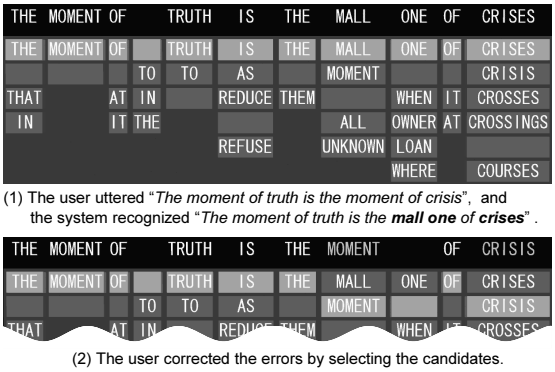

Figure 4. ***Screen snapshots of speech repair.***

3.4. System description

Figure 3 shows the architecture of our speech-repair system. Each of the six boxes in this figure represents a different process. These can be distributed over a LAN (Ethernet) and connected using a UDP-based network protocol called *RVCP (Remote Voice Control Protocol)* [9] which supports efficient multicast-based information sharing.

Acoustic signals inputted from a microphone are sent to a LAN as a packet of RVCP. The utterance detector, filled-pause detector, and acoustic feature extractor receive the packet simultaneously and output these results: the endpoints of the utterance, the beginning, and the end of each filled-pause period, and the MFCCs of the utterance, respectively. These results are received by the speech recognizer, and the recognition process is performed. At this point, the result from the filled-pause detector is used to trigger the intentional suspension function. The speech recognizer generates the confusion network as an intermediate recognition result and sends it to the interface manager. The interface manager displays the recognition results along with the competitive candidates and enables the user to select the appropriate candidate by using a device such as a mouse or a pen. Figure 4 shows an example of the output of the implemented speech-repair system.

4. Experimental results

This section describes the results of the performance evaluation of the implemented speech-repair function.

4.1. Evaluation of error correction performance

We evaluated the error correction performance of LVCSR by measuring the percentage of inclusion of correct words in the competitive candidates. The experiments were carried out on two different Japanese speech corpora: *Japanese Newspaper Article Sentences* (JNAS) as a read speech experiment, *the Corpus of Spontaneous Japanese* (CSJ) as a spontaneous speech

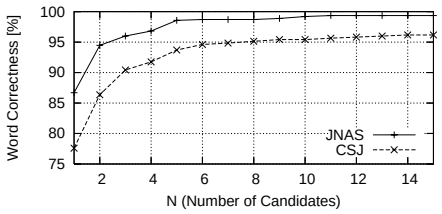

Figure 5. ***Word correctness with different number of competitive candidates displayed.***

experiment.

In the read speech experiment, we used a syllable-based HMM trained by the read speech data in JNAS and a 20 K word bigram trained by Japanese newspaper paper articles [8]. In the spontaneous speech experiment, we used a syllable-based HMM trained by the presentation speech in CSJ, and a 14K-word bigram trained by 612 lecture speech transcriptions in CSJ. As evaluation data, we used 100 sentences uttered by 20 speakers for the JNAS experiment and 100 sentences uttered by 4 speakers for the CSJ experiment. For determining the error correction performance, we evaluated word (recognition) correctness and different numbers of competitive candidates displayed. It should be noted that the normal (N = 1) recognition performance was 86.70% and 77.79% for JNAS and CSJ, respectively. Further, the number of unknown words in each evaluation data was 4 and 25 for the JNAS and CSJ, respectively.

The word correctness for each number of candidates is shown in Figure 5. It can be seen from this figure that the word correctness was drastically improved in both evaluation data. In the JNAS experiment, the word correctness finally achieved was 99.36% and this implies that the user can correct 95% of erroneous words (199/209). On the other hand, in the CSJ experiment, the word correctness finally achieved was 96.16% and this implies that 83% of erroneous words (324/391) can be corrected; however, the baseline word correctness was 10% worse than the JNAS results. In both the experiments, almost all erroneous words could be corrected even when the number of candidates displayed was five. These results show that the confusion network is sufficiently efficient and accurate to enable a user to correct erroneous words and to make the speech-repair function practical.

4.2. Usability evaluation of speech-repair system

We tested our speech-repair-enabled dictation system with 25 Japanese subjects. In order to evaluate the efficiency of the speech-repair function, we first measured the time that each subject required to enter three sentences including the error correction using our speech-repair-enabled dictation system, as shown in Figure 4. In comparison, we also tested a normal speech-dictation system that displays only a 1-best word sequence as the recognition result. In the normal speech-dictation system, erroneous words were selected using a mouse and corrected by typing on a keyboard. Next, we evaluated whether the subjects preferred to use the intentional suspension function. We measured the frequency of usage of this function under the condition that a subject could freely use it according to personal preference. After testing under both conditions, the subject was asked to complete a subjective questionnaire.

We found that our speech-repair-enabled dictation system took 31% shorter time for the sentence input compared with the case of the normal dictation system. In the second experiment, we confirmed that the intentional suspension function was used for 61% of all the sentences. The subjects tended to use this function especially for inputting long sentences. These results showed that our speech-repair function was efficient and effective for text input. The results of the questionnaire indicated that the speech-repair function was easy to use, the display of competitive candidates was helpful, and 84% of the subjects desired to use it in the future.

5. Conclusion

We have described a new speech input interface function called "*speech repair*", using which a user can easily correct errors in speech recognition by selecting candidates. In order to generate appropriate candidates and make the correction operation efficient for the user, we employ a confusion network as an intermediate recognition result. Our function enables the user to "repair" erroneous words in real time even while the user is uttering. In our experiments, almost all recognition errors were corrected and the effectiveness and usefulness of speech repair were confirmed.

In future work, we intend to further study the usefulness of the interface and the handling of unknown words and deletion error words. We also plan to apply this idea to other voice-enabled applications. The concept of building a speech interface that uses nonverbal speech information intentionally controlled by a user originated from research on "*speech completion*" [9], "*speech shift*" [10], "*speech starter*" [11], and "*speech spotter*" [12], which were followed by this research on "*speech repair*." Our future work will also aim at developing this concept further.

6. References

[1] Baumgarten, *et al*: "IBM ViaVoice QuickTutorial," *South-Western Educational Publishing*.

[2] A.E. McNair and A. Waibel: "Improving Recognizer Acceptance through Robust, Natural Speech Repair," in *Proc. ICSLP94*, pp.1299-1302, 1994.

[3] B. Suhm, *et al*: "Interactive Recovery from Speech Recognition Errors in Speech User Interface," in *Proc. ICSLP96*, pp.861-864, 1996.

[4] C-M. Karat, *et al*: "Patterns of Entry and Correction in Large Vocabulary Continuous Speech Recognition Systems," in *Proc. CHI99*, pp.568-575, 1999.

[5] L. Mangu, *et al*: "Finding Consensus in Speech Recognition: Word Error Minimization and Other Applications of Confusion Network" *Computer Speech and Language*, Vol.14, No.4, pp.373-400, 2000.

[6] M. Goto *et al*., "A real-time filled pause detection system for spontaneous speech recognition," in *Proc. of Eurospeech99*, pp. 227-230, 1999.

[7] J.Ogata, Y.Ariki: "An Efficient Lexical Tree Search for Large Vocabulary Continuous Speech Recognition," in *Proc. of ICSLP2000*, pp.967-970, 2000.

[8] T. Kawahara *et al*.,: "Recent progress of open-source LVCSR engine Julius and Japanese model repository," in *Proc. of ICSLP2004*, pp.3069-3072, 2004.

[9] M. Goto *et al*.,: "Speech Completion: On-demand Completion Assistance Using Filled Pauses for Speech Input Interfaces," in *Proc. of ICSLP2002*, pp.1489-1492, 2002.

[10] M. Goto *et al*.,: "Speech Shift: Direct Speech-Input-Mode Switching through Intentional Control of Voice Pitch," in *Proc. of Eurospeech2003*, pp. 1201-1204, 2003.

[11] K. Kitayama *et al*., "Speech Starter: Noise-robust endpoint detection by using filled pauses," in *Proc. of Eurospeech2003*, pp. 1237-1240, 2003.

[12] M. Goto *et al*.,: "Speech Spotter: On-demand Speech Recognition in Human-Human Conversation on the Telephone or in Face-to-Face Situations," in *Proc. of ICSLP2004*, pp.1533-1536, 2004.