PAPER

One-Step Error Detection and Correction Approach for Voice Word Processor*

Junhwi CHOI[†a)], **Seonghan RYU**[†b)], **Kyusong LEE**[†c)], ***Nonmembers***, ***and*** **Gary Geunbae LEE**[†d)], ***Member***

SUMMARY We propose a one-step error detection and correction interface for a voice word processor. This correction interface performs analysis region detection, user intention understanding and error correction utterance recognition, all from a single user utterance input. We evaluate the performance of each component first, and then compare the effectiveness of our interface to two previous interfaces. Our evaluation demonstrates that each component is technically superior to the baselines and that our one-step error detection and correction method yields an error correction interface that is more convenient and natural than the two previous interfaces.

key words: *speech recognition, natural language processing, languages and software systems*

1. Introduction

A voice word processor is an automatic speech recognition (ASR) system that translates speech input into the correct orthographic form of text. Even when a modern ASR system has a low error rate, the recognition results frequently include incorrect words. There have been so many studies on reducing ASR error [2]–[6], but in focus of word processor interface, an error correction process that user performs is still required to perfect a document. This correction process can be performed by selecting an erroneous portion of the text using a keyboard, a mouse, or other devices and speaking replacement text [7]–[11]. However, in some usage scenarios, error correction using only voice commands is required. For example, a handicapped person who does not have use of his or her arms may want to use only voice to correct errors. In addition, users initially tend to try to correct mis-recognized results using their own speech and often remain in the same speech modality even when faced with repeated recognition errors. Therefore, error correction using only voice commands may also be convenient for non-handicapped users.

Manuscript received November 19, 2014.
Manuscript revised April 18, 2015.
Manuscript publicized May 20, 2015.

[†]The authors are with the Department of Computer Science and Engineering, Pohang University of Science and Technology, 77 Cheongam-Ro, Nam-Gu, Pohang, Gyeongbuk, 790–784, Republic of Korea.

*The preliminary version of this paper was published in ICASSP 2012; "Seamless error correction interface for voice word processor" [1].

a) E-mail: chasunee@postech.ac.kr
b) E-mail: ryush@postech.ac.kr
c) E-mail: kyusonglee@postech.ac.kr
d) E-mail: gblee@postech.ac.kr

DOI: 10.1587/transinf.2014EDP7392

In general, conventional voice-only error correction is a two-step process: (1) the users speak a portion of the recognized text to select a target region to correct, then (2) they speak replacement text. These two steps can perform one correction. However, as Mcnair and Waibel [12] suggest, the correction process can instead be performed in a single step. In one-step correction, users speak only their replacement text, and the system automatically recognizes it correctly and finds the target region to replace. However, the conventional voice-only error correction method and the previous one-step correction method assume that the voice word processor is already in correction mode; therefore, these methods need a separated voice commands to enter and to exit correction mode [9], [12].

In this paper, we propose a one-step error detection and correction (OS-EDC) interface for a voice word processor. OS-EDC is a process like the previous one-step error correction [12], but without any explicit command to enter correction mode. Our interface automatically understands whether the intention of the current utterance is to input a new sentence or to correct a mis-recognized sentence. Then, the system detects the target region and replaces it with the current utterance. To complement the understanding of user intention, our interface can provide an optional post-confirmation interface. To demonstrate the effectiveness of our interface, we evaluate it by comparison to two voice-only error correction systems: the conventional two-step process (CTP) and the McNair & Waibel one-step process (MWOP).

To assist in understanding this paper, we define some expressions. A *current utterance* is an utterance that has not yet been classified by user intention. A *correction utterance* is a current utterance which is classified by user intention as correction. An *insertion utterance* is a current utterance which is classified by user intention as non-correction. A *target utterance* is already-typed recognition result in a document just before the current utterance and is the target of the current utterance. A *target region* is a region extracted from the target utterance, and which is can be replaced with the correction utterance if the user intention of the utterance is correction. An *error region* is a region of the target utterance, which consists of exact erroneous words. A *reference text* is intended text which the user really wants to add to the document.

The remainder of the paper is organized as follows: Sect. 2 reviews some relevant previous work. Section 3 describes our proposed method component-by-component. Section 4 describes the experimental design and presents the

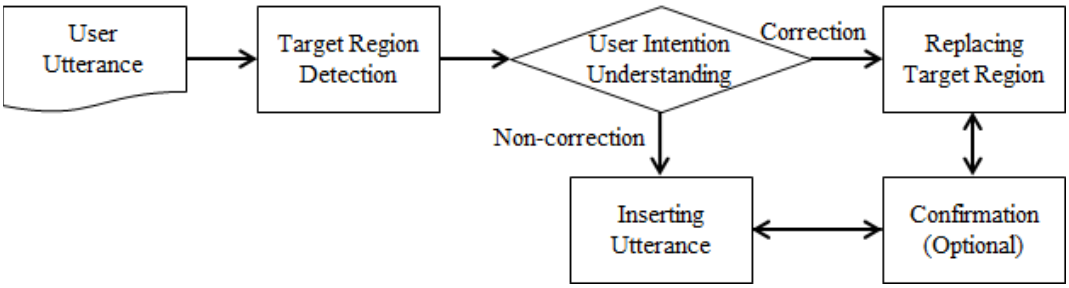

Fig. 1 Workflow of OS-EDC interface.

results of the experiments. Finally, Sect. 5 presents the conclusion of this work.

2. Previous Work

For accurate and robust OS-EDC, the system should achieve two essential functionalities: First, the target region should be accurately identified. Second, the system should understand the purpose of the current utterance. This second task corresponds to a decision to enter correction mode.

For the first functionality, Vertanen and Kristensson [13] presented three new models for automatic local aligning the target utterance with the correction utterance: a 1-best model, a word confusion network model, and a revision model. This work achieved up to 87.1% of the alignment success when aligning the target utterance with the correction utterance. When the users spoke the target region, they achieved up to 97.2% of the alignment success. However, aligning the target utterance with the target region is not suitable for one-step error correction, because the replacement text is necessary to correct target region and that process is only achievable by two-step error correction. Additionally, their work is word-based alignment and does not reflect phonetic characteristics of a correction utterance. Our OS-EDC method reflects phonetic characteristics of a correction utterance, and also considers the recognition result text surrounding the error region.

For the second functionality, previous methods such as CTP and MWOP need explicit instructions like voice commands to control the correction mode [10], [12]. CTP needs two explicit commands: a target region selection command and a replacement command. MWOP needs one explicit command which covers both target region selection and replacement in a single step. Our OS-EDC method does not need any explicit command, because the method includes a user intention understanding process.

For an entire interface, CTP and MWOP need a command so the user must speak additional words. This requirement causes inconvenience and ineffectiveness because users must remember the exact command.

3. Method

3.1 OS-EDC Scenario

When using our interface (Fig. 1), a user utters a sentence to type or correct, then our system detects the target region for accurate understanding of user intention from a target utterance which has already been typed. Then user intention understanding (i.e., the classification of the intention of the utterance as correction or non-correction) proceeds. When the intention of the current utterance is classified as correction, recognition of the utterance is revised and the detected target region is replaced automatically. Otherwise, the current utterance is inserted at the end of the document. Exceptionally, if the current utterance is the first utterance of the user, the current utterance is automatically regarded as non-correction, because there is no utterance to correct. If the user finds that the intention understanding process has committed some errors, the user can use an optional post-confirmation process. For example, if the user wants to type ‘I have an apple’, but the recognition result is ‘I have and apple’, then to correct the error region, ‘and’, the user just utters ‘an apple’. Then, the system detects the target region, ‘and apple’, and automatically determines that user intention is correction, then changes the target region ‘and apple’ changes to ‘an apple’.

In the post-confirmation process, our interface provides four commands: a target region window control command, a re-uttering command, a user intention changing command, and a cancel command. Users can adjust target region by using the target region window control command. If the correction utterance is recognized with error but user intention is classified correctly, the user can use the re-uttering command. The user intention changing command changes correction to non-correction or non-correction to correction. The post-confirmation process optionally complements the OS-EDC process for perfect document error correction.

3.2 Target Region Detection

In this step the system identifies the target region in the target utterance. From the target region, the system extracts features for user intention understanding to classify whether the intention is correction or non-correction. When the user

intention is classified as correction, the detected target region is replaced by the correction utterance.

Considering the characteristics of ASR error, the pronunciation sequence of the correction utterance and that of the target region are similar. Furthermore, for purposes of correction, without explicit instruction, users tend to speak correctly recognized text that surrounds the error region [13], [14]. The better the performance of the ASR system is, the more similar the pronunciation sequences are (we prove it in Sect. 4). Then, to detect the target region, the system locally aligns the pronunciation sequence of the current utterance to that of the target utterance. As a local alignment algorithm, we use the Smith-Waterman algorithm [15] with a mismatch penalty that is calculated from a phonetic confusion matrix [16]. We assume that the correction utterance consists of words, not pronunciations, so we detect every word included in the pronunciation sequence that is aligned by the local alignment.

In some cases, meaningless words that are caused by disfluency mostly [17], such as 'um' or 'ah', can be attached to the front or end of the recognition result. If the meaningless words are attached to the target utterance, the target region should include them. However, if only the local alignment of pronunciation sequences is considered, the target region cannot easily include these words. To solve this problem, when the previous word or next word of the target region is a meaningless word, the target region is extended based on the bi-gram score of its part-of-speech (POS)[†] label. For extension, a meaningless word set should be prepared. For bi-gram score calculation of POS label, we train POS bi-gram models by using a POS-tagged corpus which is used for the ASR language model. The meaningless words may have comparatively low POS bi-gram score in relation to the POS bi-gram scores of other words in the ASR output. Then, if the meaningless word gets the lowest score in relation to the POS bi-gram scores of the other words, the word would be included in the target region.

3.3 User Intention Understanding

The key novel process in our interface is user intention understanding. This process classifies user intention as correction or non-correction. User intention understanding can be achieved by exploiting two characteristics of correction utterances. First, user correction utterances to ASR usually have the characteristics of clear speech, which is a speaking style adopted by a speaker aiming to increase the intelligibility for a listener. To increase the intelligibility of their speech, users make on-line adjustments; typically, they speak slowly and loudly, and articulate in a more exaggerated manner [18]–[20]. Furthermore, utterances for correction display these characteristics more conspicuously than the utterances for non-correction [20]. Second, if the previous sentence has erroneous words and a user wants to correct it, then the next utterance is a correction utterance [13]. Therefore, the correction utterance is phonetically similar to a target region of the target utterance. We approach the task as a classification problem. We collect data from users, label the data with intentions, and extract and refine some of the data's features.

3.3.1 Wizard of Oz Data Collection

To classify the user's intention for the current utterance, we should collect user utterances and label them as correction or non-correction for training. Therefore, we collected data using the Wizard of Oz (WOZ) method [21], in which a human supervisor simulates the operation of our OS-EDC interface. The WOZ method is indispensable to observe the human natural behavior to the voice error correction interface [22].

Each user was required to use the system to create a document. The user was guided regarding how to correct mis-recognized results in ASR. In this process, we did not impose any prosodic characteristics on the user, and thus, we could collect data with the natural prosody of speech for correction or non-correction.

The supervisor prepared 4∼5 mis-recognized sentences from ASR for each task sentence. These prepared mis-recognized sentences are the ones that had actually occurred in ASR, because the presented situation should be a realistic representation of using the OS-EDC interface. The prepared sentences evenly included insertion errors, substitution errors, and deletion errors. Then, the supervisor listened to the user's utterances from behind the system. The supervisor showed the user a mis-recognized result on purpose to elicit a correction utterance from the user. Then, the supervisor replaced the error with the correct sentence.

We collected 910 utterances from 21 users with WOZ method. The users consist of 18 male adults and 3 female adults. All the users have no experience about using any voice word processor interface. We recorded all utterances and labeled them with labels that indicated whether the intention of the utterance was for correction or for non-correction. Then, 494 utterances are labeled as correction and 416 utterances are labeled as non-correction. Each utterance has one sentence and average number of words is 8.01 (3.12 for correction sentences and 13.82 for non-correction sentences). Using 121,000 Korean words language model, the sentences have unigram perplexity of about 268.76 and out-of-vocaburary rate of 0.02%. The overall duration of the raw wave data is about 11,844sec including silence and about 1,796sec for only voiced parts (1,035sec for correction utterances and 761sec for non-correction utterances). The average duration of the utterances is about 1.97sec (2.10sec for correction utterances and 1.82sec for non-correction utterances). The min pitch of the voiced parts is 80.51Hz and the max pitch of the voiced parts is about 494.27Hz. The average intensity of the voiced parts is about 77.45dB (78.11dB for correction utterances and 76.67dB for non-correction utterances). We refined the raw

[†]For Korean, we used Sejong POS tag set which consists of 45 tag labels.

speech data to training data to use machine-learning techniques. The training data consist of several instances. Each instance contains the features explained below and is labeled with the user's intention by human annotators. However, some features vary with each user and utterance, so we constructed the machine learning model with a focus on normalizing those features.

3.3.2 Prosodic Features

First, to classify user intention, we focus on prosodic features. Usually, prosodic features are used to classify clear speech [18]–[20]. Based on the prosodic features on the data, we found the characteristics of the clear speech in correction data rather than non-correction data. Within same user, especially the pitch range and the intensity range of the correction utterances are wider than those of the non-correction utterances and the duration per syllable of the correction utterances is larger than that of the non-correction utterances, so that we choose prosodic features including pitch, intensity and duration features.

For all users and utterances, the prosodic features should be normalized [23]. Then, our method normalize the prosodic features to the ratio of the features of a target region to those of the current utterance, where the target region is the sub-utterance of a target utterance sentence and is detected from the target region detection. Then, the system calculates the ratio of each prosodic feature between the target region and the current utterance as

$$(Feature)\ \ Ratio = \frac{(Feature)\ \ of\ \ T}{(Feature)\ \ of\ \ C}, \tag{1}$$

where T is the target region and C is the current utterance.

The ratio value represents the direction of change as well as the degree of change from the features of the target region to the features of the current utterance; therefore, the ratio produce confusion because the degree of change varies among users. So we also use 'tendency', which represents only the direction of change:

$$(Feature)\ \ Tendency = \begin{cases} 0 & (Feature)\ \ Ratio > 1 \cdot \\ 1 & Otherwise \end{cases} \tag{2}$$

This equation means that if the feature of a current utterance is larger than that of a target region, the tendency value is 1.

We separated the specific features into three categories (Table 1).

3.3.3 Distance between Pronunciation Sequences

We measure the distance between the pronunciation sequence of the target region and the pronunciation sequence of the current utterance. We use Levenshtein distance as a distance measure, but this distance depends on the length of the pronunciation sequence [24], [25]. Even when the different proportions of two pronunciation sequences are equal, the shorter pronunciation sequence has a lower Levenshtein distance value; therefore, it must be normalized as

$$Distance = \frac{LD}{l} \tag{3}$$

where LD is the Levenshtein distance value between the pronunciation sequence of target region and the pronunciation sequence of the current utterance, and l is the length of the pronunciation sequence of the current utterance. Low values indicate high similarity between pronunciation sequences.

Table 1 Normalized prosodic features for classifying user intention.

Prosodic Category	Specific Features
Pitch	Ratio of maximum pitch
	Ratio of minimum pitch
	Tendency of max pitch
	Tendency of min pitch
Intensity	Ratio of maximum intensity
	Ratio of minimum intensity
	Tendency of maximum intensity
	Tendency of minimum intensity
Duration per Syllable	Ratio of target utterance duration
	Ratio of target region duration
	Tendency of target utterance duration
	Tendency of target region duration

4. Experiments

First, we evaluate each component. Then we combine all components into an OS-EDC interface and perform quantitative and qualitative evaluation.

4.1 Target Region Detection

Accurate detection of the target region is important when erroneous words have occurred in the target utterance. Therefore, detection accuracy must be evaluated. To measure the target region detection accuracy, the system shows sentences with ASR errors to users, and the users then speak a correction utterance while selecting directly the target region that they intend to replace. We evaluate the detection accuracy by comparing the user-selected target region with the system's automatically detected target region. We evaluate full-region-unit accuracy rather than word-unit accuracy, because if the detected region does not match perfectly, it will create additional correction tasks. The full region detection accuracy equation is

$$Full\ \ Region\ \ Detection\ \ Accuracy = \frac{number\ \ of\ \ perfectly\ \ matched\ \ pairs}{number\ \ of\ \ total\ \ pairs}, \tag{4}$$

where the pair consists of the user-selected region and the system's automatically detected region and 'perfectly matched pair' means the pair which the user-selected region and the system's automatically detected region are equal.

We prepared 540 sentences with a total of about 5,800 Korean syllables, and each sentence included one or two

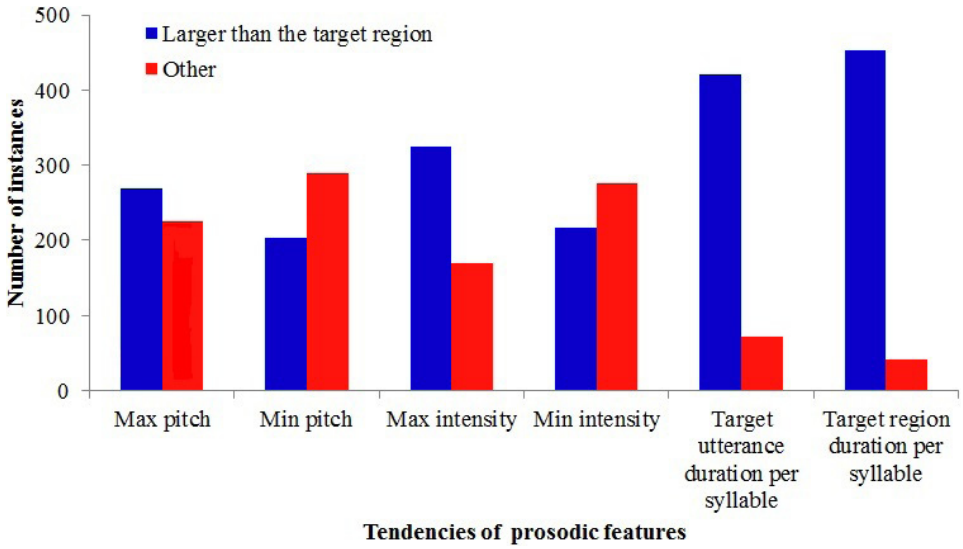

Fig. 2 Distribution of correction instances by the tendencies of prosodic features.

Table 2 Full region detection accuracy of the target region detection.

	Full region detection accuracy (%)
Baseline (word based target region detection)	84.28
CN	85.12
CN+UNK	84.33
Pronunciation sequence based target region detection	93.53
+ POS label n-gram based target region extension	93.88

ASR errors. The training data were generated by the Hidden Markov Model (HMM) based ASR system [26] for which the language model is constructed using approximately 121,000 Korean words. The users spoke corrections to correct the sentences, and we compared the region which users had selected with the region which had been detected by our method. As a baseline, we used a word based target region detection method, and also, we compared our method with the previous work of Vertanen and Kristensson [13] using confusion network model (denoted CN) and confusion network + unknown word model (denoted CN+UNK).

The best result was obtained by pronunciation sequence based method combined with the POS label n-gram based method (Table 2). In correction utterance cases, this combination can achieve 93.88% accuracy, because the average number of words in a correction utterance is about 3.12.

In case of erroneous and meaningless words such as 'um' or 'ah', is attached to the target sentence, only POS label n-gram based target region extension includes those words, because the words are not in the correction and thus the users did not respeak the words for correction.

4.2 User Intention Understanding

4.2.1 Feature Verification

Before evaluating classification accuracy, we analyzed the data to verify features that we used. We used the WOZ method to collect utterances from 21 users. We generated training data that included 910 instances from raw wave data. We labeled 416 instances as non-correction and the other 494 instances as correction. Each instance has an intention label and 13 features (12 normalized prosodic features and 1 normalized distance feature).

In correction utterances the various prosodic features differed (Fig. 2). The best-separated prosodic feature was target region duration per syllable: correction utterances tended to be spoken more slowly than the target utterances. Also, the correction utterances showed the characteristics of clear speech, in that the pitch range and intensity range were slightly wider than those of the target regions.

Correction and non-correction utterances had different distributions of normalized Levenshtein distance (Fig. 3). Correction utterances had distance values than non-correction utterances; this means that this distance feature is effective for distinguishing correction utterances from non-correction utterances.

4.2.2 Classification Accuracy

To evaluate the classification accuracy of our interface's understanding of user intention, we used a support vector machine (SVM) as a classifier, because SVM is suitable to the binary classification problem and features of our method are well-separated as shown in Sect. 4.2.1. As kernel function, we used radial basis function (RBF)[†]. We used 10-fold cross

[†]Hyper-parameter is setted with C: 1.0 and γ: 0.1.

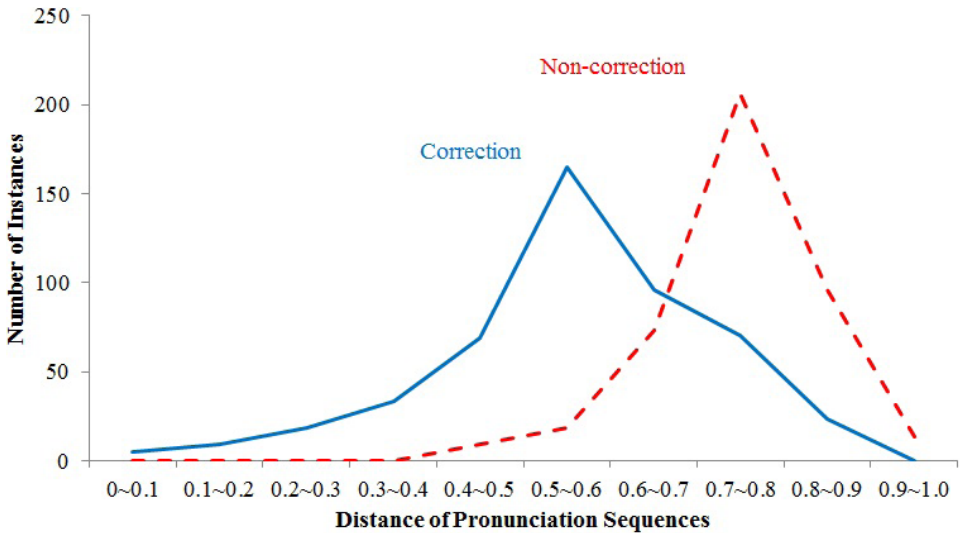

Fig. 3 Distribution of instances by the normalized Levenshtein distance between pronunciation sequences.

Table 3 User intention classification accuracy.

Features Category	Accuracy (%)
Baseline (majority)	54.29
Normalized pitch	73.27
Normalized intensity	72.12
Normalized duration per syllable	80.01
Pronunciation sequence distance	80.16
All normalized prosodic features	79.39
Normalized duration per syllable + pronunciation sequence distance	81.72
All normalized prosodic features + pronunciation sequence distance	81.90
Normalized prosodic features without ratios + pronunciation sequence distance	84.62

validation to evaluate our approach (Table 3). As a baseline, we used the majority of utterances.

In voice word processor, the false correction and non-correction caused duplicated tasks which make users uncomfortable. Therefore, accuracy is the most important evaluation value.

The best user intention understanding result was obtained by combining the tendencies of all prosodic features and the distance of pronunciation sequences; it achieved 84.62% classification accuracy. The ratio features reflected information on both the direction and the degree of change. As single normalized prosodic features, the ratio features were effective. However, in a combined model (all normalized prosodic features + distance feature), the degree of change confused the classification user intention, because the degree of change was relatively smaller than the distance; therefore, in the combined model, considering only the tendency features was more effective than all other configurations.

The most effective single feature was the normalized Levenshtein distance between pronunciation sequences; it achieved 80.16% classification accuracy, but it depended on the accuracy of the ASR. ASR with a high error rate produced many erroneous words and caused a higher normalized Levenshtein distance feature. Therefore, other features are required that are independent of the accuracy of ASR.

4.3 Overall System Evaluation

We constructed our OS-EDC interface including every component; target region detection, user intention understanding and error correction recognition. We evaluated three interfaces; CTP, MWOP, and our OS-EDC interface. CTP and MWOP, need commands to control the correction mode, so we added some commands: ‘select’ and ‘correct’ for CTP, and ‘correct’ for MWOP. For example, to correct a sentence “I have and apple” to “I have an apple”, to correct one error, the CTP interface needs two utterances, “select have and apple” for selection and “correct have an apple” for correction, and the MWOP interface needs one utterance, “correct have an apple”. For target region detection, we applied our target region detection method.

For overall system evaluation, we gave 21 users the task of making documents each consisting of 10 sentences with 264 Korean syllables. The users were the same as the users for data collection using the WOZ method, however, they had small experience about using the voice word processor interface even that the interface is for WOZ data collection. All the users had the three interfaces, and were given usage guidelines which did not include prosodic guidelines (such as “for correction, speak loudly”). The order of using interfaces was randomized for each user. After the users completed the task of making the documents, we used a questionnaire to collect the users’ qualitative assessments of the interfaces [27].

4.3.1 Quantitative Evaluation

The effectiveness is represented by quantitative evaluation (Table 4). We measured the average number of spoken syllables and the average number of turns required to complete a task. We also measured the average number of syllables and average number of turns required to correct an error. In both cases the totals included the numbers of command syllables and command turns. We used paired t-tests to assess the significance of differences noted.

Our OS-EDC interface reduced the average number syllables per task compared to the CTP interface and the MWOP interface: the differences were statistically significant ($p = 0.026$ for the CTP interface, $p = 0.024$ for the MWOP). Our OS-EDC interface also reduced the average number of turns per task compared to the CTP interface and the MWOP interface; the differences were statistically significant ($p = 0.026$ for the CTP interface, $p = 0.024$ for the MWOP interface). Compared to MWOP, our OS-EDC interface improved the effectiveness by about 21.93% for the average syllables number of per task and by about 22.30% for the average number of turns per task.

For one error correction, our OS-EDC interface reduced the average number of syllables per error compared to the CTP interface and to the MWOP interface: the differences were statistically significant ($p = 0.022$ for the CTP interface, $p = 0.009$ for the MWOP interface). Our OS-EDC interface also reduced the average number of turns per error compared to the CTP interface and the MWOP interface: the differences were statistically significant ($p = 0.018$ for the CTP interface, $p = 0.017$ for the MWOP interface). For one error correction, our interface also improved the effectiveness by about 39.55% for syllables and by about 35.97% for turns compared to the MWOP interface.

This result shows that our interface can work effectively with 93.88% detection accuracy for analysis region detection, 84.62% classification accuracy for user intention understanding.

In case of a sentence with two or more erroneous words, the OS-EDC interface is more efficient than other interfaces. When the users used the OS-EDC interface, the users tended to correct erroneous words one-by-one with short utterances. Contrastively, when the users used the other interfaces, the user tended to correct erroneous words at once that means respeak the whole sentence with command, and it caused supplementary corrections because of the arise of another erroneous word in recognition results of the respeaking. However, OS-EDC interface uses prosodic and pronunciation sequence features in the user intention understanding phase, so that the utterances which provide so much insufficient information such as just one word for correction are not classified well. In particular, as cases of the correction of the word is a noun, users tend to respeak just the word and duration per syllable of the utterance is even shorter than target. That phenomenon appears to the early time of conducting the task of making document and is surely reduced when the user adapted to the OS-EDC interface.

4.3.2 Qualitative Evaluation

We measured user convenience level and intuitiveness level by using questionnaire survey on which the users scored each item on a scale of 1 to 10 (Table 5). The convenience level represents how users rate the convenience of use each interface, and reflects some inconveniences, such as long utterances for error corrections and false operations of the interfaces. The intuitiveness level represents users' assessment of how easy each interface is to use.

The proposed interface received higher assessments for both convenience and intuitiveness of use (Table 5). All differences were statistically significant: $p = 0.011$ for convenience level of the CTP interface, $p = 0.019$ for convenience level of the MWOP interface, $p = 0.027$ for intuitiveness level of the CTP interface, $p = 0.009$ for intuitiveness level of the MWOP interface. Our method achieved scores of 8.64/10 for convenience level and 8.36/10 for intuitiveness level and those scores are higher than the scores of the previous two methods. In convenience level, an interface that has more effectiveness scores higher. In intuitiveness level, the CTP interface scored higher than the MWOP interface because the CTP interface is a fully explicit correction method for selection and correction using two commands, but the MWOP interface operates selection implicitly and correction explicitly, so the MWOP interface can confuse users. However, our OS-EDC interface is a fully implicit correction method and is also natural in that it follows the method that humans use when correcting errors in a dictation task.

Both users who used the OS-EDC interface first and the others rated higher the OS-EDC interface than the other interfaces. But, one remarkable distinction between users who used the OS-EDC interface first and the others is that the users who used the OS-EDC interface first make mistakes omitting commands when they used the CTP and MWOP interfaces for experiment. However, the users using the CTP or MWOP interfaces first did not use any commands during using the OS-EDC interface. Additionally, they rated the convenience level and the intuitiveness level of the OS-EDC interface higher than the other interfaces and the score difference is more than the other users' score difference. This

Table 4 Effectiveness of three correction interfaces.

	Syllables	Turns	Syllables per error	Turns per error
CTP	390.18	22.78	35.74	3.62
MWOP	363.77	17.55	24.24	1.83
OS-EDC	320.01	14.35	17.37	1.34

Table 5 Qualitative evaluation of three correction interfaces.

	Convenience level	Intuitiveness level
CTP	6.27	7.91
MWOP	8.33	7.02
OS-EDC	8.64	8.36

phenomenon can be an evidence of convenience and intuitiveness of the OS-EDC interface.

5. Conclusion

We have proposed a One-Step Error Detection and Correction (OS-EDC) interface for a voice word processor. OS-EDC includes target region detection, user intention understanding and error correction utterance recognition. We detect the target region by using a pronunciation-sequence-based method with a POS label n-gram-based region extension, and achieved 93.88% accuracy in full-region detection. We classify user intention using normalized prosodic features and a normalized Levenshtein distance feature, and achieved 82.91% accuracy. Furthermore, using all components combined, we proved the effectiveness of our OS-EDC interface by comparison to two previous methods (CTP interface and MWOP interface).

The OS-EDC interface understands whether a user's new utterance is intended to be a correction or non-correction. By understanding a user's intention, we provide a system that can automatically enter correction mode in response to only the correction utterance of a replacement text. Furthermore, the OS-EDC interface is humanlike, because it replicates the human habit of correcting mistakes during conversations. The mistakes include speaking mistakes and understanding mistakes. With this method, when composing a document, users need not remember any voice commands for entering correction mode and may simply speak sentences that they want to type. Therefore, the efficiency of the process may be increased. In addition, the developers of a voice word processor need not design any voice commands for entering correction mode.

Acknowledgments

This work was partly supported by ICT R&D program of MSIP/IITP [B0101-15-0307, Basic Software Research in Human-level Lifelong Machine Learning (Machine Learning Center)] and also was partly supported by National Research Foundation of Korean [NRF-2014R1A2A1A01003041, Development of multi-party anticipatory knowledge-intensive natural language dialog system].

References

[1] J. Choi, K. Kim, S. Lee, S. Kim, D. Lee, I. Lee, and G.G. Lee, "Seamless error correction interface for voice word processor," 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp.4973–4976, IEEE, 2012.

[2] K. Larson and D. Mowatt, "Speech error correction: The story of the alternates list," International Journal of Speech Technology, vol.6, no.2, pp.183–194, 2003.

[3] M. Jeong, S. Jung, and G.G. Lee, "Speech recognition error correction using maximum entropy language model," Proc. INTERSPEECH, pp.2137–2140, 2004.

[4] W.K. Seong, J.H. Park, and H.K. Kim, "Dysarthric speech recognition error correction using weighted finite state transducers based on context-dependent pronunciation variation," Computers Helping People with Special Needs, Lecture Notes in Computer Science, vol.7383, pp.475–482, Springer, Berlin, Heidelberg, 2012.

[5] H. Cucu, A. Buzo, L. Besacier, and C. Burileanu, "Statistical error correction methods for domain-specific ASR systems," Statistical Language and Speech Processing, Lecture Notes in Computer Science, vol.7978, pp.83–92, Springer, Berlin, Heidelberg, 2013.

[6] M. Hacker and E. Noth, "A phonetic similarity based noisy channel approach to ASR hypothesis re-ranking and error detection," 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp.2317–2321, IEEE, 2014.

[7] J. Sturm and L. Boves, "Effective error recovery strategies for multimodal form-filling applications," Speech Commun., vol.45, no.3, pp.289–303, 2005.

[8] D. Huggins-Daines and A.I. Rudnicky, "Interactive ASR error correction for touchscreen devices," Proc. 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Demo Session, pp.17–19, ACL, 2008.

[9] K. Vertanen and P.O. Kristensson, "Parakeet: A continuous speech recognition system for mobile touch-screen devices," Proc. 14th International Conference on Intelligent User Interfaces, pp.237–246, ACM, 2009.

[10] A. Kumar, T. Paek, and B. Lee, "Voice typing: A new speech interaction model for dictation on touchscreen devices," Proc. SIGCHI Conference on Human Factors in Computing Systems, pp.2277–2286, ACM, 2012.

[11] Y. Liang, K. Iwano, and K. Shinoda, "Simple gesture-based error correction interface for smartphone speech recognition," Fifteenth Annual Conference of the International Speech Communication Association, 2014.

[12] A.E. McNair and A. Waibel, "Improving recognizer acceptance through robust, natural speech repair," Third International Conference on Spoken Language Processing, 1994.

[13] K. Vertanen and P.O. Kristensson, "Automatic selection of recognition errors by respeaking the intended text," IEEE Workshop on Automatic Speech Recognition & Understanding, ASRU 2009, pp.130–135, IEEE, 2009.

[14] M. Leijten and L. Van Waes, "Keystroke logging in writing research: Using inputlog to analyze and visualize writing processes," Written Communication, vol.30, no.3, pp.358–392, 2013.

[15] T.F. Smith and M.S. Waterman, "Identification of common molecular subsequences," Journal of Molecular Biology, vol.147, no.1, pp.195–197, 1981.

[16] D. Han and K. Choi, "A study on error correction using phoneme similarity in post-processing of speech recognition," The Journal of the Korea Institute of Intelligent Transport Systems, pp.77–86, The Korean Institute of Intelligent Transport Systems (Korean ITS), 2007.

[17] S. Goldwater, D. Jurafsky, and C.D. Manning, "Which words are hard to recognize? Prosodic, lexical, and disfluency factors that increase speech recognition error rates," Speech Commun., vol.52, no.3, pp.181–200, 2010.

[18] S.-J. Moon and B. Lindblom, "Interaction between duration, context, and speaking style in English stressed vowels," The Journal of the Acoustical Society of America, vol.96, no.1, pp.40–55, 1994.

[19] R. Smiljanic and A. Bradlow, "Production and perception of clear speech in Croatian and English," The Journal of the Acoustical Society of America, vol.116, no.4, p.2627, 2004.

[20] E. Godoy, M. Koutsogiannaki, and Y. Stylianou, "Approaching speech intelligibility enhancement with inspiration from Lombard and clear speaking styles," Computer Speech & Language, vol.28, no.2, pp.629–647, 2014.

[21] J.F. Kelley, CAL—A natural language program developed with the OZ paradigm: Implications for supercomputing systems, IBM Thomas J. Watson Research Center, 1985.

[22] N. Dahlbäck, A. Jönsson, and L. Ahrenberg, "Wizard of OZ studies: Why and how," Proc. 1st International Conference on Intelligent

User Interfaces, pp.193–200, ACM, 1993.
[23] A. Stolcke, S. Kajarekar, and L. Ferrer, "Nonparametric feature normalization for SVM-based speaker verification," IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2008, pp.1577–1580, IEEE, 2008.
[24] W.J. Heeringa, Measuring dialect pronunciation differences using Levenshtein distance, Ph.D. thesis, Citeseer, 2004.
[25] L. Yujian and L. Bo, "A normalized Levenshtein distance metric," IEEE Trans. Pattern Anal. Mach. Intell., vol.29, no.6, pp.1091–1095, 2007.
[26] X.D. Huang, Y. Ariki, and M.A. Jack, Hidden Markov models for speech recognition, Edinburgh University Press, Edinburgh, 1990.
[27] M.Q. Patton, Qualitative evaluation and research methods, SAGE Publications, inc, 1990.

Junhwi Choi is a Ph.D. student at the Department of Computer Science and Engineering at POSTECH, Pohang, South Korea. He received his B.S. degree at the Department of Computer Science and Engineering at POSTECH, Pohang, South Korea. His research interests include ASR error correction, ASR adaptation, and robust dialog management.

Seonghan Ryu is a Ph.D. student at the Department of Computer Science and Engineering at POSTECH, Pohang, South Korea. He received his B.S. degree at the Department of Computer Science and Engineering at Dongguk University, Seoul, South Korea. His research interests include multi-domain spoken dialog system, robust dialog management, and exploiting web resources.

Kyusong Lee is a Ph.D. student at the Department of Computer Science and Engineering at POSTECH, Pohang, South Korea. He received his B.S. degree at the Department of Computer Science and Engineering at Soongsil University, Seoul, South Korea. His research interests include grammar error correction, dialog-based computer assisted language learning, and student modelling.

Gary Geunbae Lee received his B.S. and M.S. degrees in Computer Engineering from Seoul National University in 1984 and 1986 respectively. He received Ph.D. degree in Computer Science from UCLA in 1991 and was a research scientist in UCLA from 1991 to 1991. He has been a professor at the CSE department, POSTECH in Korea since 1991. He is a director of the Intelligent Software laboratory which focuses on human language technology researches including natural language processing, speech recognition/synthesis, and speech translation. He authored more than 100 papers in international journals and conferences, and has served as a technical committee member and reviewer for several international conferences such as ACL, COLING, IJCAI, ACM SIGIR, AIRS, ACM IUI, Interspeech-ICSLP/EUROSPEECH, EMNLP and IJCNLP. He is currently leading several national and industry projects for robust spoken dialog systems, computer assisted language learning, and expressive TTS.