SPEECH-TO-TEXT INPUT METHOD FOR WEB SYSTEM USING JAVASCRIPT

Ryuichi Nisimura[1], *Jumpei Miyake*[2], *Hideki Kawahara*[1], *and Toshio Irino*[1]

[1]Faculty of Systems Engineering, Wakayama University, Japan
[2]Graduate School of Information Science, Nara Institute of Science and Technology, Japan

ABSTRACT

We have developed a speech-to-text input method for web systems. The system is provided as a JavaScript library including an Ajax-like mechanism based on a Java applet, CGI programs, and dynamic HTML documents. It allows users to access voice-enabled web pages without requiring special browsers. Web developers can embed it on their web page by inserting only one line in the header field of an HTML document. This study also aims at observing natural spoken interactions in personal environments. We have succeeded in collecting 4,003 inputs during a period of seven months via our public Japanese ASR server. In order to cover out-of-vocabulary words to cope with some proper nouns, a web page to register new words into the language model are developed. As a result, we could obtain an improvement of 0.8% in the recognition accuracy. With regard to the acoustical conditions, an SNR of 25.3 dB was observed.

Index Terms— Speech recognition, Speech communication, Internet, User interfaces

1. INTRODUCTION

We have developed a speech recognition interface to realize useful voice-enabled web systems. It is provided as a JavaScript library comprising a sophisticated mechanism based on a Java applet, CGI programs, and Ajax-like dynamic HTML documents. We can easily introduce a speech-to-text input method to all web pages. On the client side, we can access the voice-enabled websites using basic browsers without any add-on programs. The users can use a basic PC and web browsers such as Internet Explorer, Mozilla Firefox, and Safari. Therefore, our method can be easily used by web developers, editors, and bloggers who may not have technical expertise. They can embed our interface by inserting only one line of code in their HTML document. It is not required for both the users and the developers to prepare speech recognition programs by using our public shared Japanese speech recognition server (ASR server).

This study also aims at observing natural human-machine spoken interactions in home and office environments. A large-scale voice database is indispensable for the study of voice interfaces. We have been collecting voices in cases such as simulated scenarios, WOZ (Wizard of Oz) tests[1], telephone-based systems[2, 3], and public space tests[4]. However, an evaluation of voices captured by normal PCs is still insufficient to organize a voice database for personal use. In order to develop and utilize the speech interface further, it is important to know the recording conditions prevalent in home and office environments[5]. Further, it is also important to know the configurations of the PCs used for recording voices. In order to realize the actual data collections, the proposed method has a remarkable design suitable for the field testing of the spoken interface system. Our speech recognition engine server has gathered all input voices using our JavaScript library over the Internet. We have succeeded in collecting 4,003 inputs in a period of seven months from real users. The collected voices are tested by using our speech recognizer to investigate the actual conditions during web use.

1.1. Voice-enabled web systems

Many advanced web systems that enable voice-based interactions have been proposed in the past. Voice Extensible Markup Language (VoiceXML)[1], which is the W3C (World Wide Web Consortium)[2] standard XML format, can be used to develop a spoken dialogue interface with a voice browser[6]. SALT (Speech Application Language Tags)[3] is also an extension of HTML that adds a speech interface to web pages[7]. They require special add-on programs or customized web browsers. Therefore, users must preinstall a speech recognition and synthesis engine and prepare nonstandard programs for voice recording. However, their complicated setup programs make it difficult for users to access voice-enabled websites.

The WebGalaxy system[8][9] is a Java-based system. Although it can be used to realize a spoken interface on web pages, web developers need to study speech technologies in order to develop voice-enabled web pages.

In order to obtain a large amount of actual voice data, low cost and effort for users and developers are important in introducing spoken interface. Our method can offer a suitable design for performing field tests of spoken interface systems in home and office environments. It is also practical and reasonable with regard to develop voice-enabled websites.

2. SPEECH-TO-TEXT JAVASCRIPT LIBRARY

2.1. How to use (User's viewpoint)

Figure 1 shows voice-enabled web pages that use our method. Our method offers a speech-to-text input method for free text input areas by using the "`<INPUT TYPE="text" ...>`" and "`<TEXTAREA> ... </TEXTAREA>`" HTML elements.

The recording panel is opened when a user double clicks on the text input area. The panel that opens upon clicking is a dynamic HTML program written in the JavaScript language. It also contains a button implemented as a pure Java applet that records the voice of a user. When the recording panel is activated by pressing the button, the voice of a user can be recorded by the microphone of a client PC. After the button is released, the button applet begins to upload the recorded signals to the ASR server. The button applet also provides visual feedback to the user. During voice recording, a bar of the button becomes a power level meter. The users can monitor their voice signal level by observing the red bar.

[1]http://www.voicexml.org/
[2]http://www.w3.org/
[3]http://www.saltforum.org/

978-1-4244-3472-5/08/$25.00 ©2008 IEEE

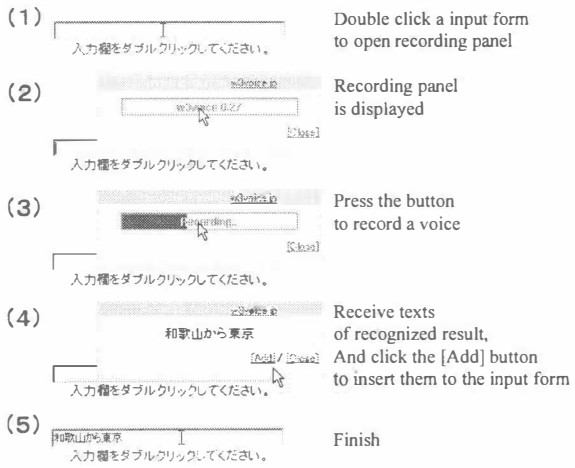

Fig. 1. *Close-up of speech-to-text input area on web pages.*

The ASR server recognizes the received voice and outputs the transcribed text in an XML format to the JavaScript program. Then, the recording panel displays the transcribed text after the transmission is completed. Finally, the text is confirmed by the user by clicking the [Add] button on the panel. The confirmed text is inserted into the text area. It is reasonable to expect that web services such as a search engine, transit information, and language translation will be capable of accepting voice-inputted text via the web browser.

2.2. How to install (Developer's viewpoint)

Figure 2 shows the HTML code used to load the proposed interface. An additional setting for web developers is to only add the underlined part (<script ... ></script ... >) to the header field of their HTML document. Our system automatically enables a speech-to-input method to all text input areas.

We have named the library "w3voiceIM.js," and its current version is distributed at the following address.

```
http://w3voice.jp/engine/w3voiceIM.js
```

Web developers can directly insert the above address into their HTML documents without downloading any files to their local / server computers.

2.3. Mechanism

A brief architecture of the proposed method is shown in Figure 3. The system comprises three independent computers connected through the Internet: client PC, Web server, and ASR server. The client PC and Web server are typical components in a web system. In addition, for realizing our method, the shared ASR server is added; it communicates with the client PC via the signal transmission protocol based on the HTTP (Hypertext Transfer Protocol).

On the client PC, clicking on a text input field by a user invokes the recording panel comprising a dynamic HTML program (JavaScript) and the button applet. The button applet is a key component of our voice-enabled web framework. It is a Java applet (using Java Sound API of standard Java APIs) that records the voice of a user and controls the entire proposed mechanism. The button applet has been implemented such that it functions as an upload routine similar to a web browser program. After a user's voice is captured,

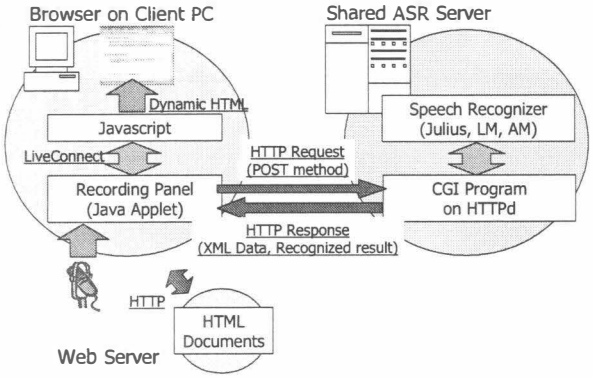

Fig. 3. *Software architecture.*

the button applet begins to transmit the data to the ASR server using the HTTP request protocol (POST method). The voice data saved in the ASR server is converted to text data immediately by the speech recognition module. The recognized text data is communicated to the button applet by the XML data in the HTTP response protocol. Then, the button applet sends the received text data to the JavaScript program using the LiveConnect[4] protocol. As a result, the text is displayed on the recording panel by the dynamic HTML program.

The shared ASR server is implemented as an application of CGI programs running on a HTTP (Web) server. It includes a speech recognition module comprising a decoder, a trigram language model, and an HMM acoustic model. Because the voice signals obtained from the button applet are raw audio data (PCM, 16 bits, 44.1 kHz sampling rate), the module converts signals to MFCCs parameter for transcribing the text with the decoder.

We have designed a server-side architecture on the basis of open and standard technologies using free software. All message exchanges are performed over standard internet protocols such as the HTTP. In other words, users can utilize the proposed method from any location over a broadband network. Our system is compatible to deal with the fire wall system in an office network.

3. PUBLIC FIELD-TEST OVER THE INTERNET

We released the proposed library ("w3voiceIM.js") on July 3, 2007 on our project's website[5]. All recorded inputs from users were stored in the ASR server in order to analyze the actual conditions of spoken interactions in web use. We have succeeded in collecting 4,003 inputs during a period of seven months (from July 1, 2007 to January 28, 2008) from real users.

Figure 4 plots the number of inputs per day. The number of maximum accesses was 786 that had been recorded at the news release[6] date. As shown in the figure, the system has succeeded in acquiring data from a constant number of users. The number of unique IP addresses of the users was 530. In other words, we have successfully acquired information through the public testing of our system from users who repeatedly used our system.

Figure 5 shows the distributions of utterance lengths, where the horizontal axis indicates the lengths of the inputs (s.). In this figure, 731 (24.3%) zero-length inputs, which occurred due to a single click

[4]http://en.wikipedia.org/wiki/LiveConnect

[5]http://engine.w3voice.jp/engine/ (in Japanese)

[6]http://slashdot.jp/articles/07/07/09/0715202.shtml (News release in Japanese)

```
<html>
<head>
<script language="javascript" type="text/javascript" src="http://w3voice.jp/engine/w3voiceIM.js"> </script>
<title>Text page</title>
</head>
<body>
.....
</html>
```

Fig. 2. *An example of HTML documents. <script> element is added to load the JavaScript library (underlined part).*

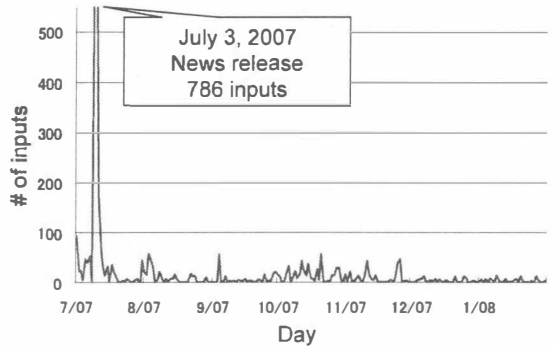

Fig. 4. *Number of inputs per day.*

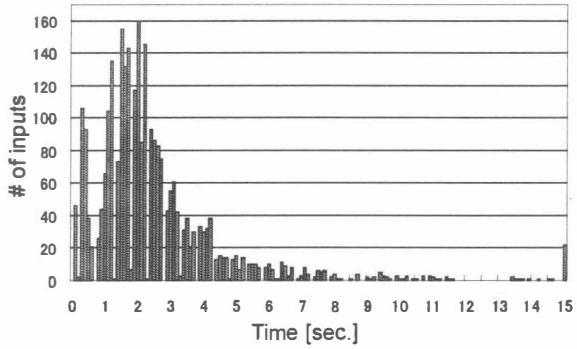

Fig. 5. *Distribution of input time lengths. ($0 < s < 15$)*

of the button applet, are excluded. Further, long inputs having a length of over 15 seconds are not shown. The average length without zero-length inputs was 2.6 seconds. It was shown that the voices in web use usually comprised short sentences.

4. EVALUATIONS

In evaluations, we tested the speech recognition rates using the collected real voices. We also examined the SNR (Speech-to-Noise Ratio) of the recorded signals in order to investigate the recording conditions of a user's environments.

4.1. Test data

With regard to the test data, 496 voices were extracted from the collected data, which were recorded from July 10 to August 23, 2007. We excluded zero-length inputs from the test data. They were manually transcribed to transcriptions of user utterances, and segmented into speech and silence parts to compute the SNR on the basis of the signal powers. The number of morphemes in the transcriptions was 2,330, which implies that the sentences had 4.7 words per utterance.

4.2. ASR setup

We have developed our Japanese speech recognition program, which is the same as the ASR server used in daily operations.

An open source speech recognition engine Julius ver.3.5.3[10] is adopted as the decoder. As for an HMM acoustic model, we used a speaker independent phonetic tied-mixture (PTM) triphone model, which was trained from reading style speech in clean environments.

In order to cope with the wide linguistic variation encountered in web use, an original Japanese trigram language model was developed from the following text collections:

- Web texts gatherd by an automatic web crawler (3881k sentences, 39939k words).
- Japanese Wikipedia[7] texts (4077k sentences, 110102k words).
- List of train station names in Japan (235k sentences, 1038k words, 9310 stations).

In total, our collection contained 526,726 different words. Among these, a lexical vocabulary with 50,418 words was organized by choosing frequently occurring words (70 times or more in the case of nouns, 400 times or more in the case of others).

4.3. Results

Figure 6 shows the distributions of word correct rates and SNRs. The y-axis indicates the average word correct rate [%], while the x-axis indicates the averaged SNR [dB] of each utterance.

In total, the word correct rate was 43.0% among 496 utterances. The trigram testset perplexity was 49.8, and an out-of-vocabulary rate against the test data was 4.1%.

We included 210 data (42.3% of 496 utterances) whose word correct rate is 0%. These were usually caused by the use of some proper nouns in limited expressions. In the next breath, 157 utterances with a recognition rate of 100% also existed. We can say that the insufficient recognition accuracy is due to mismatches of linguistic characters between the sentence and language models.

We could not observe a significant correlation between recognition rates and SNRs. The total average of the SNR was 25.3 dB. Moreover, there were few utterances that were made with very bad conditions in a noisy environment. The results prove that improving

[7] http://ja.wikipedia.org/

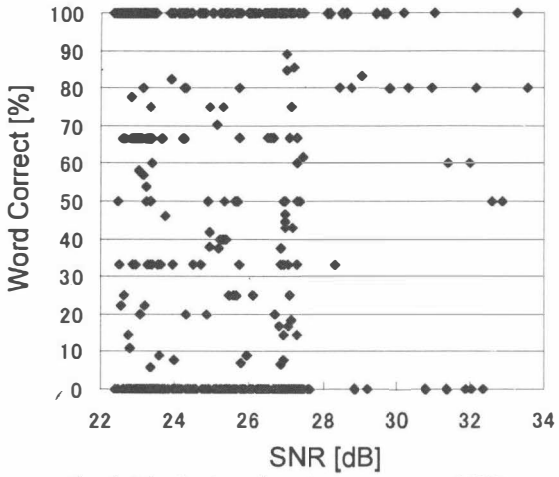

Fig. 6. *Distribution of word correct rate and SNR.*

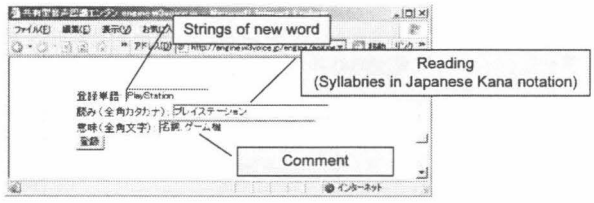

Fig. 7. *Web page for unknown word registrations*

the linguistic performance is necessary to improve speech recognition accuracies. For example, specifying the linguistic task domain by referring to the web content would lead to conduce good accuracies.

5. UNKNOWN WORD REGISTRATIONS

As shown in the evaluation results, a linguistic mismatch was one of the main causes of the insufficient ASR performance. In particular, the influence of unknown words in the language model appears to be noticeable. In order to address this problem, our ASR server provides a web page (Figure 7) where users can register new words into the language model. Items that can be entered on this page are listed include:

- Strings,
- Reading (Japanese syllabaries),
- Comments.

New word entries from among these items are added to the model with a probability value that is reserved for the unknown word class (<UNK>). "Strings" is a character string used for lexical entries. "Reading" would be converted to pronunciation symbols by using the Japanese Kana-to-Romaji routine. "Comments" will be used for semantic annotations in the future.

We have obtained 58 new word entries from the users in the field-test of our public ASR server. Then, experimental results showed that an improvement of 0.8% in the recognition accuracy was obtained by the addition of these new words in the model vocabulary. We can say that the advantage of unknown word registrations via the web page was confirmed although there were some adverse effects such as increased insertion errors due to the addition of short words (e.g. "Wii"). For improving the language models, it is necessary to develop a method for dynamic vocabulary restructuring in order to add unknown words provided by the users.

6. CONCLUSIONS

A speech-to-text JavaScript library for web systems is proposed based on an Ajax-like mechanism comprising a Java applet, CGI programs, and dynamic HTML pages. It enables users to access voice-enabled web pages without requiring special browsers and add-on programs. Notable advantages for web developers include the ease of installing our system into a web page. Our public ASR server has gathered a large number of input voices in order to observe natural human-machine spoken interactions in personal environments. We have succeeded in collecting 4,003 inputs during a period of seven months. A web page used for registering unknown words conduced 0.8% improvements of recognition accuracy. An investigation of the acoustical conditions revealed an SNR of 25.3 dB.

In future studies, we plan to introduce linguistic domain controls according to the contents of web pages. In order to realize this, a task estimation algorithm for a web page will be investigated.

7. REFERENCES

[1] N. Kawaguchi et al., "Multimedia Corpus of In-Car Speech Communication," *Journal of VLSI Signal Processing-Systems for Signal, Image, and Video Technology*, vol.36, pp.153–159, 2004.

[2] A. Raux, et al., "Doing Research on a Deployed Spoken Dialogue System: One Year of Let's Go! Experience," *Proc. Interspeech2006*, pp. 65–68, 2006.

[3] M. Turunen, et al., "Evaluation of a Spoken Dialogue System with Usability Tests and Long-term Pilot Studies: Similarities and Differences." *Proc. Interspeech2006*, pp. 1057–1060, 2006.

[4] R. Nisimura et al., "Operating a Public Spoken Guidance System in Real Environment". *Proc. Interspeech2005*, pp.845–848, 2005.

[5] S. Hara et al., "An online customizable music retrieval system with a spoken dialogue interface", The Journal of the Acoustical Society of America (4th Joint Meeting of ASA/ASJ), Vol. 120, No. 5, Pt. 2, pp.3378–pp.3379, Nov. 2006.

[6] S. McGlashan et al., "Voice Extensible Markup Language (VoiceXML) Version 2.0", *W3C Technical Reports and Publications*, W3C, 2004.

[7] "SALT: Speech Application Tags (SALT) 1.0 Specification", the SALT Forum, 2002.

[8] R. Lau et al., "WebGalaxy - Integrating Spoken Language and Hypertext Navigation". *Proc. EUROSPEECH97*, pp.883–886, 1997.

[9] A. Gruenstein et al., "Scalable and Portable Web-Based Multimodal Dialogue Interaction with Geographical Database". *Proc. Interspeech2006*, pp.453–456, 2006.

[10] A. Lee et al., "Julius – An Open Source Real-Time Large Vocabulary Recognition Engine", *Proc. EUROSPEECH2001*, pp.1691–1694, 2001.