PodCastle: Recent Advances of A Spoken Document Retrieval Service Improved by Anonymous User Contributions

Masataka Goto and Jun Ogata

National Institute of Advanced Industrial Science and Technology (AIST)
1-1-1 Umezono, Tsukuba, Ibaraki 305-8568, JAPAN

Abstract

In this paper, we introduce recent advances of a speech retrieval web service, *PodCastle*, that collects and amplifies voluntary contributions by anonymous users. Our goal is to provide users with a public web service based on speech recognition and crowdsourcing so that they can experience state-of-the-art speech recognition performance through a useful service. PodCastle enables users to find speech data (such as podcasts and *YouTube* video clips) that include a search term, read full texts of their recognition results, and easily correct recognition errors by simply selecting from a list of candidates. The resulting corrections were used to improve both the speech retrieval and recognition performances. In our experiences from its practical use over the past four years (since December, 2006), over half a million recognition errors in about one hundred thousand speech data were corrected by anonymous users and we confirmed that the speech recognition performance of PodCastle was actually improved by those corrections.

Index Terms: speech retrieval, spoken document retrieval, wisdom of crowds, crowdsourcing

1. Introduction

Since the amount of speech data available on the web is ever-increasing in the form of podcasts (audio blogs), individual audio files, video clips such as those on video sharing services like *YouTube*, etc., there are growing needs for information retrieval for speech data. Unlike text data, however, the speech data themselves cannot be used as an index for information retrieval. Although metadata or social tags are often put on speech data, such metadata annotated by someone, such as categories or topics, tend to be broad and imperfect and are not sufficient for deep, useful information retrieval. To achieve full-text retrieval based on spoken content in speech data [1, 2, 3, 4, 5, 6, 7], we need to use automatic speech recognition (ASR) for text transcription. We therefore launched a speech retrieval web service called *PodCastle* [8, 9, 10, 11, 12, 13] [1] in 2006 that provides full-text searching of speech data available on the web on the basis of ASR technologies, and have been developing and improving its functions since then. PodCastle allows web users to find speech data that include a search term, and read full texts of their recognition results.

Given the speech data on the web, however, it is difficult to achieve the ASR with high accuracies. The ASR technologies cannot avoid making recognition errors for various types of contents in speech data. Because of the diversity of topics, vocabularies, and speaking styles, speech corpora covering its diversity could not be prepared in advance. It is also difficult to support new words and phrases, such as proper names and buzzwords, often used in speech data published recently. Out-of-vocabulary words are therefore inevitable in recognizing web-based speech data. As a result, it is not easy to provide a spoken document retrieval service with high accuracies. Users of such a web service might be disappointed by ASR performances.

[1] Demonstration video clips of the PodCastle service are available at http://staff.aist.go.jp/m.goto/PodCastle/ .

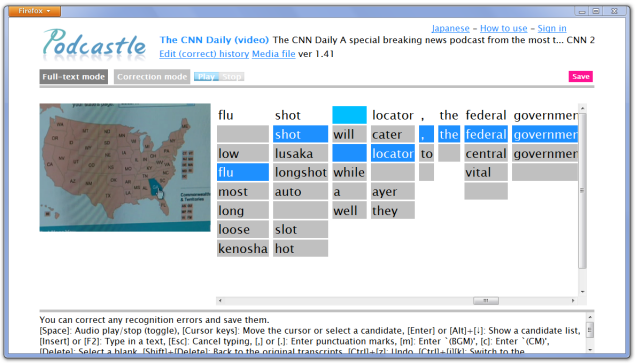

Figure 1: *PodCastle screen snapshot of an interface for correcting speech recognition errors (competitive candidates are presented underneath the normal recognition results). Three errors in this excerpt were corrected by selecting from the candidates.*

To overcome these difficulties, our PodCastle web service enables anonymous users to contribute by correcting speech-recognition errors. PodCastle provides the full text of speech recognition results for podcasts, individual audio or movie files on the web, and video clips on video sharing services *YouTube*, *Nico Nico Douga*, and *Ustream.tv*. Users can read those full texts with a cursor moving in synchronization with the audio playback on a web browser. If a user finds a recognition error while listening, the user can easily correct the error by simply selecting from a list of candidates or typing the correct text on an efficient error correction interface [14] shown in Figure 1. The resulting corrections can then be used not only to immediately be shared with other users and improve the spoken document retrieval performance for corrected speech data, but also to gradually improve the speech recognition performance by training our speech recognizer so that other speech data can be searched more reliably. This approach can be described as *collaborative training for speech recognition*.

Furthermore, we have developed a mechanism of automatic learning of ever-increasing new words and phrases from news articles and dictionaries on the web. Since most current ASR technologies recognize only words in static system vocabularies, out-of-vocabulary words are wrongly recognized as a combination of existing words. On the other hand, PodCastle automatically collects new words and phrases, their pronunciation, and usage examples from web pages [9]. The collected words and phrases are then added to the ASR system vocabulary and the context in which they are used (i.e., n-gram) is also learned. PodCastle can thus catch up with fresh contents on the web.

PodCastle was released at *http://podcastle.jp* on December 1st, 2006 as a public system that uses the *wisdom of crowds* or *crowdsourcing* for speech processing. So far, 765 speech programs such as podcasts and YouTube channels have been registered, consisting of 112,476 audio files in total (as of May 31st, 2011). Of these, 2,593 audio files have been at least partially corrected, which resulted in 521,938 corrected words (errors). We found some podcast programs registered in PodCastle were corrected almost everyday or every week, and confirmed the performance improvement by the wisdom of crowds.

2. Overview of PodCastle

PodCastle is a social annotation web service where users can search, read, and annotate web speech data in text form. PodCastle originally supported only audio podcasts, and has then been extended to support video podcasts, individual audio or video files on the web, and video clips on a video sharing service *YouTube*, a video communication service *Nico Nico Douga*, and a video streaming service *Ustream.tv*. Each podcast consists of a series of episodes of audio data (MP3 files) and their metadata (RSS syndication feed). Each YouTube channel also consists of a series of video clips having audio data and has its RSS. A podcast or a YouTube channel is denoted as the term *speech program*, which mentions a group of speech data. With RSS, updated speech data (episodes or video clips) are automatically downloaded from the web.

Just as full-text search services are essential for accessing text web pages, there is a growing need for full-text speech retrieval services such as PodCastle. Although there were previous research projects for speech retrieval [1, 2, 3, 4, 6, 7] before 2006, most do not provide public web services for podcasts. There were two major exceptions, Podscope [15] and PodZinger [16], which started web services for speech retrieval targeting English-language podcasts in 2005. Our PodCastle was the first service in 2006 to provide full-text searching of Japanese-language podcasts. Even if the other earlier services could also support Japanese speech data, they only displayed parts of speech recognition results, making it impossible to visually ascertain the detailed contents of the speech data and to search them using other existing text-based search engines. Even if their users find that search results are degraded by unavoidable speech recognition errors, the users had no means of correcting these errors. In contrast, PodCastle allows full-text results of speech recognition to be accessed by both users and external search services, and allows a number of users to contribute to improve the speech retrieval/recognition performance.

2.1. Three Functions of PodCastle

PodCastle supports three functions of searching, reading, and annotating speech data. The searching and reading functions let users better understand the speech recognition performance, and the annotating (error correction) function allows them to contribute to improved performance. This improved performance can then lead to a better user experience of searching and reading speech data.

2.1.1. Searching Function

This is a function that allows a full-text search of speech recognition results. When the user types in a search term, a list of speech data containing this term is displayed together with text excerpts of speech recognition results around the highlighted search term. These excerpts can be played back individually.

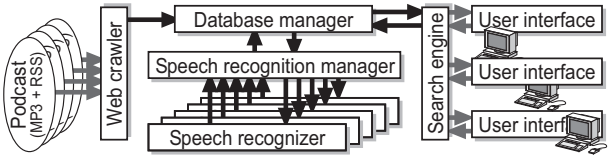

Figure 2: *Implementation overview of PodCastle.*

By selecting one of these search results, the user is then able to access its full text by switching over to the reading function.

2.1.2. Reading Function

With this function, as well as listening to a speech data the user can also view the transcribed text of the speech data. To make errors easy to discover, each word is colored according to the degree of reliability estimated during speech recognition. Furthermore, a cursor moves across the text in synchronization with the audio playback. Because the full-text result of speech recognition being applied to each speech data becomes available to external full-text search engines, such results can be discovered together with ordinary web pages by those engines.

2.1.3. Annotating Function (Error Correction)

This function allows users to add "annotations" to correct any recognition errors. Here, annotation means transcribing the contents of speech data, either by selecting the correct candidate from the list of competitive candidates, or by typing in the correct text. For this purpose, we provide an efficient error correction interface we earlier proposed [14] (Figure 1). A recognition result excerpt is shown around the cursor and scrolled in synchronization with the audio playback. Each word in the excerpt is accompanied by other word candidates, which are generated beforehand by using a *confusion network*[2] [17] that can condense a huge internal word graph of a large vocabulary continuous speech recognition (LVCSR) system. Note that users are not expected to correct all the errors, but can be expected to correct some errors according to their interests.

3. Implementation of PodCastle

The implementation overview of PodCastle is shown in Figure 2. The *web crawler* collects speech data, which can be added by users. Those speech data are then recognized by multiple *speech recognizers* one after another. When a request from a speech recognizer is received by the *speech recognition manager*, the next available speech data to be recognized is handed over. After the recognizer finishes processing its speech data, the recognition result is passed to the database manager via the speech recognition manager. The *database manager* controls the processing state of the speech data and indexes their speech recognition results together with the corrections provided by users. Finally, the *search engine* works as a website that provides the PodCastle *user interface*. Because this architecture allows speech recognizers to be executed anywhere in the world, PodCastle could serve as a research platform where other researchers can also exhibit their own recognition results. This will be especially useful for supporting various languages.

The web server was implemented by using a web applica-

[2]The confusion network was originally introduced in the context of word error minimization, which minimizes the word error rate of recognition results rather than the sentence error rate [17]. Our original idea is to apply such an efficient intermediate recognition result to generate competitive candidates for efficient error correction [14].

tion framework *Ruby on Rails*, a programming language *Ruby*, a web server *Passenger* and *Apache*, a database *MySQL*, a morphological parser for Japanese language *ChaSen*, and a full-text search engine *Senna* and *Tritonn*. The client interface was implemented by using a scripting language *JavaScript* and its library *MochiKit*, multimedia frameworks *QuickTime* and *Flash*, and ActionScript compilers *MTASC* and *Flex*.

To recognize speech data, audio data is first segmented into three categories — speech, music without speech, and other background sounds — by applying GMMs. Speech segments are then recognized by using our in-house LVCSR system based on an efficient N-best search algorithm [18] to generate the confusion networks. This system uses cross-word tied-state triphone HMMs trained for 39-dimensional PLP-based features, and a 243k-word trigram language model trained by using both large standard corpora and daily-updated web news.

We had to overcome several difficulties in recognizing various speech data on the web. In terms of language modeling, for example, web speech data tend to include words and phrases related to recent topics, which are usually not registered in the system vocabulary. We therefore developed a method to keep a language model up-to-date [9] by using online dictionary entries obtained from a dictionary site *Hatena Keyword* and online news texts obtained from a news site *Yahoo! News*. Note that dictionary entries on the Hatena Keyword are also added and maintained by anonymous users, i.e., by the wisdom of crowds. In addition, in terms of acoustic modeling, web speech data include various types of acoustic conditions. To reduce the acoustic mismatch, we apply several improvement methods such as iterative unsupervised adaptation.

The automatic performance improvements through user corrections were achieved through various techniques, such as those for training an acoustic model by using corrected transcriptions, for making a language model adapt to different topics by using metadata and corrected transcriptions, and for registering out-of-vocabulary words by using a phonetic typewriter to estimate their pronunciation. The details are described in [9, 10].

4. Experiences with PodCastle

PodCastle was released to the public at *http://podcastle.jp* on December 1st, 2006. In addition to audio podcasts, PodCastle has then supported video podcasts in August 2009 and video clips on *YouTube*, *Nico Nico Douga*, and *Ustream.tv* (recorded videos) in 2011 by transcribing speech data in video clips and displaying an accompanying video screen in synchronization with the original PodCastle screen as shown in Figure 1. PodCastle has also recently supported functions of annotating speaker names and paragraphs (new lines), marking (changing the color of) correct words that do not need any correction, and showing the percentage of correction, which becomes 100% when all the words are corrected or marked as "correct". When several users are correcting different parts of the same speech data, those corrections can also be automatically shared (synchronized) and shown on their screens. This is useful for seriously and rapidly transcribing speech data together.

As shown in Figure 3, as of May 31st, 2011, 765 Japanese speech programs such as podcasts and YouTube channels have been registered, consisting of 112,476 audio files in total. Of these, 2,593 audio files have been at least partially corrected, which resulted in 521,938 corrected words (errors) as shown in Figure 4. We found that 52.8% were corrected by the candidate selection, whereas 47.2% were corrected by the text typing.

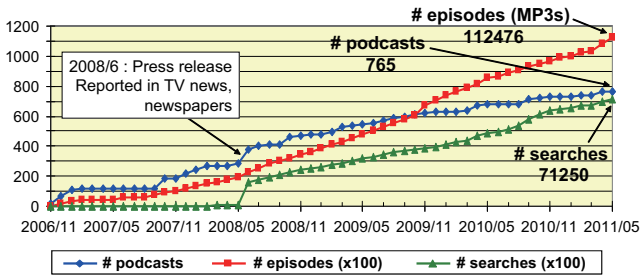

Figure 3: Cumulative usage statistics of PodCastle: the number of podcasts, the number of episodes (audio or video files), and the number of searches (queries).

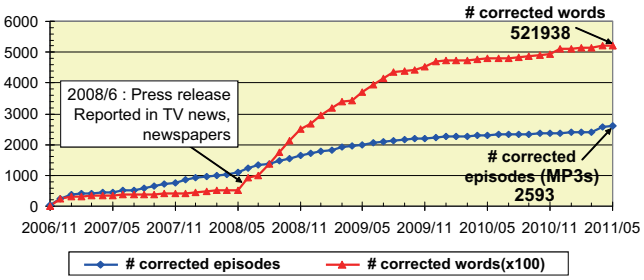

Figure 4: Cumulative usage statistics of PodCastle: the number of corrected episodes and the number of corrected words.

After the public press release on June 12th, 2008, PodCastle was reported in television and newspapers, which increased the use of PodCastle as shown in Figures 3 and 4. We found that there are users who voluntarily cooperate in the correction, as happens with other Web 2.0 services, and that speech data recorded by famous artists and TV personalities tend to receive many corrections. Some podcasts were corrected almost everyday or every week. Here, note that we did not expect to obtain complete corrections. Rather than expecting a few users to make large contributions, we solicit small contributions from large numbers of users.

For the collaborative training of our speech recognizer, we introduced a podcast-dependent acoustic model that is trained for each podcast using its transcripts corrected by anonymous users [9, 10, 11]. Through our experiments, we confirmed that the speech recognition performance for some podcasts that received many error corrections was actually improved by the acoustic model training (relative error reduction of 21-33%) [10] and the burden of error correction was reduced for those podcasts. Furthermore, we confirmed that the performance was also improved by the language model training, which will be reported in another paper in the future.

We have inferred some motivations for users correcting speech recognition errors on PodCastle, though we cannot directly ask since the users are anonymous. These motivations can be categorized as follows:

- *Error correction itself is enjoyable and interesting*
  Since the error correction interface is carefully designed to be useful and efficient, using it, especially for quick and accurate operations by proficient users, could be a form of fun somewhat like a video game.
- *Users want to contribute*
  Some users would often correct errors not only for their own convenience, but also to altruistically contribute to better speech recognition and retrieval.

- *Users want their speech data to be correctly searched*

  The creators of speech data (like podcasters for podcasts) would correct recognition errors in their own speech data so that it can be more accurately searched.
- *Users like the content and cannot tolerate the presence of recognition errors in it*

  Some fans of famous artists or TV personalities would correct errors because they like the speakers' voices and cannot tolerate the presence of recognition errors in their favorite content. In fact, we have observed that such speech data generally receive more corrections than other types.

One Web 2.0 principle is to trust users and we also trust users with respect to the quality of correction. In fact, as far as we assessed the quality, the correction results obtained so far have been of high quality. One of the reasons would be that PodCastle avoids relying on monetary reward like Amazon Mechanical Turk. Even if some users deliberately make inappropriate corrections (vandalism), though, we will be able to develop countermeasures to acoustically evaluate the reliability of corrections. For example, we can use the likelihood of HMMs for forced alignment with the corrections.

5. Conclusion

We have described recent advances of the PodCastle web service that provides a search engine for web speech data on the basis of the wisdom of crowds or crowdsourcing. The technical contribution of this study is to investigate how far the performance of speech recognition and full-text search can be improved by getting recognition errors corrected through the cooperative efforts of many end users. This is a *social correction* framework, where users can share their correction results over a web service. The game-based approach of Human Computation or GWAPs (games with a purpose) [19] like the ESP Game [20] often depends on the feeling of fun and lacks the feeling that the improved performance leads to a better user experience which encourages further use of the service. In this framework, users gain a real sense of contributing to the convenience of themselves and other users, and can be motivated to contribute by seeing the correction activities made by other users.

Another contribution of this study was that it demonstrates how speech recognition can be put to use in situations where a speech corpus is almost impossible to prepare in advance. Although speech recognition usually requires a sufficient corpus to provide useful results, such corpora tend to be costly and labor-intensive, thus limiting applications. On the other hand, this study has aimed at *collaborative training for speech recognition* where full-text transcriptions containing recognition errors are first disclosed and then corrected by anonymous users. Since there are many errors, we run the risk of attracting criticism, but we believe that sharing these results with users will promote further popularization and use of speech recognition.

Furthermore, we think this study provided a framework for *amplifying* user contributions. In a typical Web 2.0 service like *Wikipedia*, improvements were limited to an item directly contributed (edited) by users. In PodCastle, improvements are automatically spread to other items not contributed by users because of the improvement of the speech recognition performance. This is a novel technology of amplifying user contributions, which is beyond the Web 2.0 and the Human Computation [19]. We hope that this study will prove the importance and potential of incorporating and amplifying user contributions, and that various other projects [21, 22] that follow this approach will be done, thus adding a new dimension to this field of research.

6. Acknowledgments

We thank Youhei Sawada, Shunichi Arai (Mellowtone Inc.), Kouichirou Eto (AIST), and Ryutaro Kamitsu (Brazil Inc.) for their web service implementation. We also thank anonymous users of PodCastle for correcting speech recognition errors.

7. References

[1] S. Whittaker, *et al.*, "SCAN: Designing and evaluating user interfaces to support retrieval from speech archives," in *Proc. of ACM SIGIR 99*, 1999, pp. 26–33.

[2] J.-M. V. Thong, *et al.*, "Speechbot: An experimental speech-based search engine for multimedia content on the web," *IEEE Trans. on Multimedia*, vol. 4, no. 1, pp. 88–96, 2002.

[3] L. Lee and B. Chen, "Spoken document understanding and organization," *IEEE Signal Processing Magazine*, vol. 22, no. 5, pp. 42–60, 2005.

[4] V. Turunen, M. Kurimo, and I. Ekman, "Speech transcription and spoken document retrieval in Finnish," *Machine Learning for Multimodal Interaction*, vol. 3361, pp. 253–262, 2005.

[5] J. Besser, K. Hofmann, and M. Larson, "An exploratory study of user goals and strategies in podcast search," in *Proc. of FGIR Workshop Information Retrieval (WIR 2008)*, 2008, pp. 27–34.

[6] Cambridge Multimedia Document Retrieval Project, http://mi.eng.cam.ac.uk/research/projects/mdr/.

[7] CMU Informedia Digital Video Library Project, http://www.informedia.cs.cmu.edu/.

[8] M. Goto, J. Ogata, and K. Eto, "PodCastle: A Web 2.0 approach to speech recognition research," in *Proc. of Interspeech 2007*, 2007, pp. 2397–2400.

[9] J. Ogata, M. Goto, and K. Eto, "Automatic transcription for a Web 2.0 service to search podcasts," in *Proc. of Interspeech 2007*, 2007, pp. 2617–2620.

[10] J. Ogata and M. Goto, "PodCastle: Collaborative training of acoustic models on the basis of wisdom of crowds for podcast transcription," in *Proc. of Interspeech 2009*, 2009, pp. 1491–1494.

[11] ——, "PodCastle: A spoken document retrieval system for podcasts and its performance improvement by anonymous user contributions," in *Proc. of SSCS 2009*, 2009, pp. 37–38.

[12] M. Goto and J. Ogata, "**[Invited talk]** PodCastle: A spoken document retrieval service improved by user contributions," in *Proc. of KJDB 2010*, 2010.

[13] ——, "**[Invited talk]** PodCastle: A spoken document retrieval service improved by anonymous user contributions," in *Proc. of PACLIC 24*, 2010, pp. 3–11.

[14] J. Ogata and M. Goto, "Speech Repair: Quick error correction just by using selection operation for speech input interfaces," in *Proc. of Eurospeech 2005*, 2005, pp. 133–136.

[15] Podscope, http://www.podscope.com/.

[16] PodZinger, http://www.podzinger.com/.

[17] L. Mangu, E. Brill, and A. Stolcke, "Finding consensus in speech recognition: Word error minimization and other applications of confusion networks," *Computer Speech and Language*, vol. 14, no. 4, pp. 373–400, 2000.

[18] J. Ogata and Y. Ariki, "An efficient lexical tree search for large vocabulary continuous speech recognition," in *Proc. of ICSLP 2000*, 2000, pp. 967–970.

[19] L. von Ahn, "Games with a purpose," *IEEE Computer Magazine*, vol. 39, no. 6, pp. 92–94, June 2006.

[20] L. von Ahn and L. Dabbish, "Labeling images with a computer game," in *Proc. of CHI 2004*, 2004, pp. 319–326.

[21] N. Ramzan, *et al.*, "The participation payoff: Challenges and opportunities for multimedia access in networked communities," in *Proc. of ACM MIR 2010*, 2010.

[22] S. Luz, M. Masoodian, and B. Rogers, "Supporting collaborative transcription of recorded speech with a 3D game interface," in *Proc. of KES 2010*, 2010.