Speech-As-Data Technologies for Personal Information Devices

Roger C.F. Tucker, Marianne Hickey, Nick Haddock[1]
Client and Media Systems Laboratory
HP Laboratories Bristol
HPL-2002-42
March 18th , 2002*

E-mail: roger_tucker@hp.com, marianne_hickey@hpl.hp.com, nick@hickhaddock.com

speech-as-data, speech recognition, wordspotting, speech compression, audio summarisation

For small, portable devices, speech input has the advantages of low-cost and small hardware, can be used on the move or whilst the eyes & hands are busy, and is natural and quick. Rather than rely on imperfect speech recognition we propose that information entered as speech is kept as speech and suitable tools are provided to allow quick and easy access to the speech-as-data records. This paper summarises the work carried out at HP Labs and its partners at Cambridge University on the technologies needed for these tools - for organising, browsing, searching and compressing the stored speech. These technologies go a long way towards giving stored speech the characteristics of text without the associated input problems.

* Internal Accession Date Only
Approved for External Publication
[1] Consultant working with HP

Abstract

For small, portable devices, speech input has the advantages of low-cost and small hardware, can be used on the move or whilst the eyes & hands are busy, and is natural and quick. Rather than rely on imperfect speech recognition we propose that information entered as speech is kept as speech and suitable tools are provided to allow quick and easy access to the speech-as-data records. This paper summarises the work carried out at HP Labs and its partners at Cambridge University on the technologies needed for these tools - for organising, browsing, searching and compressing the stored speech. These technologies go a long way towards giving stored speech the characteristics of text without the associated input problems.

Keywords: Speech-as-data, speech recognition, wordspotting, speech compression, audio summarisation

1. Introduction

For small, portable devices, speech input has the advantages of low-cost and small hardware, can be used on the move or whilst the eyes & hands are busy, and is natural and quick. The limited memory and CPU resources of a small low power device can be overcome by piping the speech or speech parameters [1] to a network resource that performs speech recognition and sends the results back to the device. If a network connection is temporarily unavailable, the speech can be stored and converted to text once the connection becomes available.

However, it is not quite as easy as that. Unconstrained large vocabulary recognition requires considerable time and effort by a human to check and correct the results. It is bad enough to do this at the time of entry, much worse to have to do it later if no network connection is available. Only restricted-domain input such as calendar entries can be recognized well enough not to need much correction - products like Dragon's NaturallySpeaking® Mobile rely on this to process voice entries offline when the device is docked. In general, with current speech recognition technology, speech is in fact a very awkward way of entering textual information. But the alternatives of tiny soft-keyboards or character-based handwriting fit badly with the concept of highly portable, use-on-the-move devices, leaving this aspect of I/O essentially unsolved.

One solution is that speech is used for entering information, but that the information is retained in speech form instead of (or as well as) attempting to convert it to text, so the speech *is* the data. This has the advantage that if the information needs to be accessed eyes/hands free it is already in the right format - it also allows levels of expression that are difficult to convey in text. There are big drawbacks though. One is that a greater amount of storage is needed for speech compared to text. But the major problem is that browsing and accessing stored speech is potentially tedious and time-consuming, as anyone using today's telephone-based voice messaging systems well knows. This is actually more to do with the legacy of tape and other continuous media than anything intrinsic in the data type. With appropriate organisation and quick and effective search tools, information stored as speech can be accessed quickly and easily.

In contrast to earlier speech-as-data work concentrating on audio-only interfaces [2][3] our primary focus has been on devices with small displays such as the WinCE-based Palm-PCs. This provides a limited amount of visualisation and browsing, as well as a greater degree of organisation. In this paper we overview the work at HP Labs and its partners at Cambridge University over the last few years, showing how speech can be organised, retrieved and compressed efficiently within the limited resources of a low-power device. The aim is to utilize network-based speech recognition when it is available, but not to rely on it for our basic functionality.

2. Organising Speech Data

2.1 Speech-Enabling of Applications

Crucial to the use of speech-as-data is the ability to fill any field in any application with speech as well as text. To illustrate how this might work, we have designed a speech enabled Contacts application, see Figure 1. The names are entered by hand using the soft-keyboard, or entered as speech at the time of capture and corrected/entered later by hand. Everything else can potentially be entered and retained as speech. There are four criteria for choosing speech rather than text:

1. Do I need this information for visually browsing the records? If so, it is well worth the effort of entering text.

2. Is this information better expressed in speech or text? Personal feelings, perhaps even spoken by the person whose entry it is, are more easily conveyed using speech.
3. Is this information that would take a long time to enter by hand e.g. an address or note? Speech saves time now at the expense of possibly more time later.
4. Is this information that I may not need, but would like to keep just in case? Fax numbers, addresses and titles are fields in this category. In this case it is easy to read them in and then they are there if ever needed.

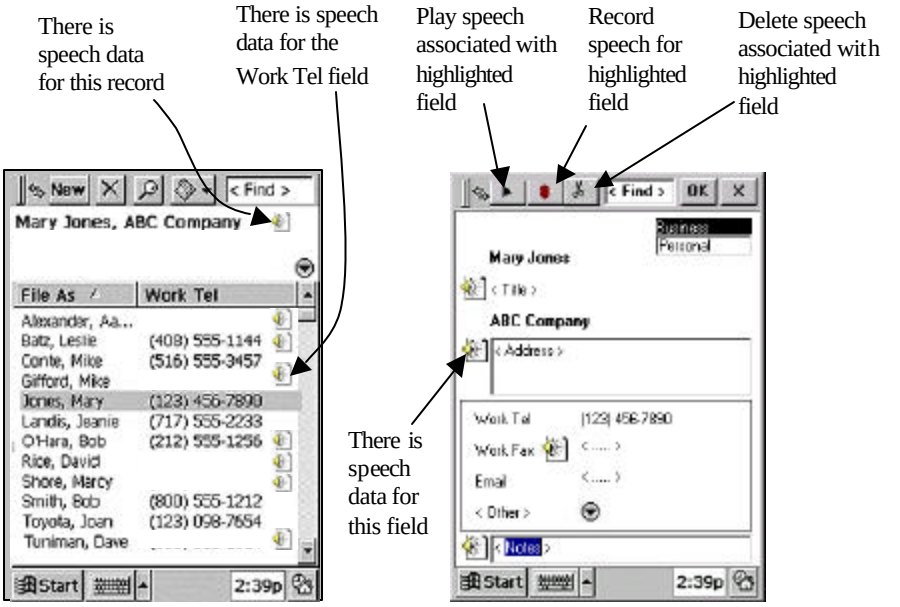

Figure (1): Speech-enabled Contacts Application

2.2 Eyes-busy entry of information

To benefit from the portability of the device, it must be possible to fill in these fields in an eyes-free way. We've done this by recording a voice memo structured with keyword labels [4]. The keywords can be pre-defined or user defined, but in either case they mark the start of a section of speech that should be cut out and inserted into a particular field. For example, whilst driving you may see something on the side of a truck and record:

> *"Contacts entry … Work Number is 408 927 6353 … Name is Hamilton & Co Removals … Comment is the truck says something about extra heavy items a specialty, could get that piano shifted at last"*

The recording is then processed in three stages:

1. The very first keyword is found, and used to file the recording into the right application.
2. Subsequent keywords are found and used to segment the recording, each segment being sent to the appropriate field in the desired application.
3. Where possible, the speech data is recognised and entered as text.

Since the entry name is not recognised, the entry will not be alphabetically placed, but just sit at the top of the Contacts list until at least the name is transcribed. To allow the detection of keywords to be done reliably and with a simple recogniser, the user is required to pause before (and possibly after) each keyword phrase. By requiring the pause to be a long one, e.g. more than one second, non-keyword phrases can be rejected quite easily as long as the pause detection is robust.

3. Browsing Speech Data

3.1 Graphical Interface

Whilst ideally all speech should be recorded in small easily accessible chunks, this will not always be possible - not all information to be recorded will fit neatly into the available fields of an application. Once more than 10 seconds or so of speech is recorded in one go, navigating becomes an important issue. A simple but effective solution is to provide a 2-D random-access interface [5]. Figure 2 shows such an interface developed at HP Labs [6], where a 100 second recording is displayed as a series of lines about 8 seconds long.

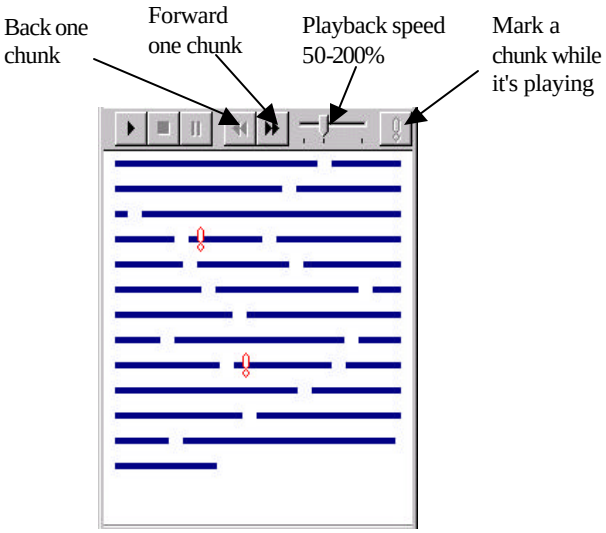

Figure (2): Random-Access Interface for Longer Speech Records

The speech record is divided into chunks based on the pauses in the recording. Whilst ideally the chunks would match phrases in the speech, it is equally important that they are a suitable length for skipping and marking. The algorithm creating the chunks [7] has the concept of a target length (about 5 seconds), which it tries to achieve within the constraints of not breaking up a continuous phrase. The ability to mark a place in the speech is very important, and used properly ensures that most of the speech need only be listened to once. A number of different markers could be made available for different categories - we have used icons for phone numbers and dates/times to good effect.

3.2 Textual Summaries

So far we have not considered the possible role of large vocabulary speech recognition if it were available, either on the client device or on a network server. This allows complete speech records to be transcribed automatically, but with a substantial percentage of errors especially for unconstrained speech. One approach is to leave these for the user to cope with, as with the AT&T SCAN interface for speech records [8]. Our approach is to *partially* transcribe the speech. The aim is to give enough of a textual summary to enable the user to quickly get to the part of the speech data they need. The user can then listen to get the full information and if they want to, complete or correct the transcription process.

Two components are used to provide the summary - an acoustic confidence score for each word, and an inverse frequency score. Other features such as prosody [9] can also be used. The acoustic confidence score can be derived a number of ways, though in the case of the ANN/HMM hybrid recogniser [10] we used, the confidence scores were obtained directly from the phone probabilities [11]. The inverse frequency score is the number of times a word appears in a recording divided by the number of times it occurred in the language model and normalised by the length of the recording. The two scores are combined:

$$\omega * \text{inverse frequency} + \alpha * \text{acoustic confidence} \qquad (1)$$

where the relative weighting factors ω and α depend on the importance of accuracy versus coverage.

The summary is created by first automatically transcribing the recording using a large vocabulary recogniser, and then scoring each word according to Equation 1. Then, an N-word window (typically 10-30 words long) is slid over the transcript, and a score for each N-gram calculated. A single parameter, *number of words per minute*, determines both the choice of N and how many N-grams are displayed. Table 1 shows some partial transcriptions of a one-minute section of the May 22, 1996 broadcast of CNN EPN. Full details of the system are given in [12] along with evaluation results.

Size	**Summary of one minute of speech**
2 keywords per minute	FERRY, OVERCROWDED
1 10-gram per minute	FERRY SINKING IN WHICH HUNDREDS OF PASSENGERS DIED [SIL] THE FERRY
1 30-gram per minute	INTO YESTERDAY'S FERRY SINKING IN WHICH HUNDREDS OF PASSENGERS DIED [SIL] THE FERRY WENT DOWN ON LAKE VICTORIA WITHIN SIGHT OF SHORE [SIL] THE GOVERNMENT SAYS FIVE HUNDRED FORTY NINE PEOPLE ON
20 (potentially overlapping) 10-grams per minute	: [SIL][SIL] AFTER A FOUR-DAY HEARING IN FEDERAL COURT A JUDGE HAS STRIPPED … YESTERDAY'S FERRY SINKING IN WHICH HUNDREDS OF PASSENGERS DIED [SIL] THE FERRY WENT DOWN ON LAKE VICTORIA WITHIN SIGHT … THIRD CLASS PASSENGERS [SIL] CRUISE TICKETS ONLY GAVE THEM A PLACE TO SIT OR STAND ON DECK [SIL] THOSE … PASSENGERS [SIL] BUT HUNDREDS MORE IGNORED WARNINGS AND JAMMED ABOARD THE VESSEL [SIL] TO MAKE THE TRIP [SIL] FUMBLE

Table (1): Example Partial Transcriptions

4. Searching Speech Data

4.1 Keyword Searching

Because browsing speech is not as fast as text, a good search facility is even more important than for text. A search facility that uses spoken queries also provides a generic eyes-free way of accessing any of the data on the device. Searches can specify a field e.g. search the "File As" field in Contacts for a specific name, or can be a free search through all the stored speech in the system. Matching one isolated utterance against another (as in the first case) is quite fast and accurate using conventional isolated word recognition techniques, but spotting for a phrase embedded in audio is both slow and difficult since the endpoints are not known. Whilst there is a large body of research on wordspotting, our task is different to a conventional wordspotting application. Normally, a single utterance is searched for a number of words for which good models exist, and the task is primarily to determine which word was spoken. In our case we are searching a potentially large number of utterances for one word that we have just one spoken example of, and the task is to decide if it is there or not. The two technical issues raised by these differences are how to use the spoken keyword in the wordspotter, and how to make the search go fast enough.

The Video Mail Retrieval (VMR1) database [13] is a database of 35 speakers recorded specifically for testing this kind of wordspotting system. As well as a number of read-speech training sentences, each speaker recorded 20 spontaneous speech messages and then 35 isolated keywords frequently occurring in the messages. Performance is measured in terms of the percentage of hits scoring higher than the Nth false alarm (or percentage misses scoring lower). Exactly which value of N is most meaningful is quite difficult to decide as it depends on how fast the hits can be assessed by the user, how much audio there is, and how many false alarms they would be prepared to listen to before finding the right one. To get around this difficulty of choosing an operating point, a metric known as the Figure of Merit (FoM) [14] can be used, which in essence gives the percentage hits averaged over 1 to 10 false alarms per hour. To give an idea of the performance of state-of-the-art speech recognition on this task, Table 2 shows the FoM for the different keywords in the VMR database, using the Abbot recognition system [10] as the wordspotter. For these results, the keywords have been entered as text and the phonetic transcription looked up in a

dictionary. Whereas the longer more complex words can be found quite reliably, the short words are easily missed.

FoM Range	Words (lowest FoM first)
Below 30%	WORD
30-40%	MAIL, DATE, VIDEO, SCORE, OUTPUT
40-50%	FIND, BADGE, ACTIVE
50-60%	TIME
60-70%	KEYWORD, STAFF, SENSOR, SPOTTING, SEARCH
70-80%	RANK, MESSAGE, INDIGO, DOCUMENT, PLAN, DISPLAY
80-90%	PROJECT, ASSESS, MEETING, PANDORA,
90-100%	WORKSTATION, MANAGE, MICROPHONE, INTERFACE, NETWORK, CAMERA

Table (2) Figure of Merit Scores for Keywords in VMR database

The next two sections address the two key issues for wordspotting in the context of audio search: how to model the spoken keyword, and the speed of the search.

4.2 Modeling the Keyword

The problem here is that the search query will have the word spoken in isolation and probably emphasized, but the utterance with it in will have it embedded in continuous speech. We therefore cannot use the query speech directly, but instead need to pass it through a phoneme recogniser to convert it to a string of phonemes. The keyword for the search is then modeled by concatenating the appropriate phoneme models.

In [15], Knill and Young investigate a number of enhancements to this basic approach, two of which give a small improvement. The first enhancement is to use the N-best phoneme strings from the phoneme recogniser in the search. Since this tends to increase the false alarms as well as the true hits, it only improves performance in the cases where the true phoneme string wasn't the most probable. Around 3% overall reduction in the number of missed words is achieved with N=7. This rather small improvement is at the expense of a slower search, as there are now N versions of the keyword to look for. The second enhancement is to use two examples of the keyword rather than one. If the two phoneme strings generated are different, both can be used, though it is as good just to use the one which gives the best Viterbi score when force-matched against the other spoken example. This achieves around 10% reduction in the number of missed words. Disappointingly, even the best spoken keyword system increased the number of missed words by a factor of at least 30% compared to using a dictionary-derived phoneme string for the search word. Nevertheless, a quick search and good user interface can make up for some lack of performance, and the user can easily improve the search by using longer words or even phrases.

4.3 Speed of Search

We would ideally like the search to go fast enough to appear instantaneous. But unlike a textual search, where all the results can be displayed and assimilated by the user very quickly, with audio each hit has to be listened to. Even if 5 seconds of speech is enough context to check a potential hit, it will take the user 25 seconds to check the top 5 hits. This time can be incorporated into the search by playing the most probable hit found so far whilst the search is taking place. As an example let us assume that there is a total of 30 minutes of speech to be searched and that in 5 seconds 7.5 minutes of speech can be searched. The hits delivered to the user could be:

1. *after 5 seconds* - best hit in the first 7.5 minutes
2. *after 10 seconds* - best hit in the first 15 minutes not already played
3. *after 15 seconds* - best hit in the first 22.5 minutes not already played
4. *after 20 seconds* - best hit in whole 30 minutes not already played

5. *after 25 seconds* – next best hit in whole 30 minutes not already played

As the first three hits are from partial searches, this strategy is not perfect, but it requires 90 times real-time rather than the 360 times real-time needed to do the whole 30 minutes in the first 5 seconds.

To achieve such a fast search on the modest computing platform we expect to find in a portable device, the speech has to be pre-processed when, or soon after, it is recorded. Fortunately, the structure of a wordspotting system lends itself to this.

A standard wordspotting system performs two recognition passes - one with keywords and fillers together, the other with just the fillers. The second pass allows the keyword scores to be normalised against speaker and acoustic variations. The fillers can simply be phoneme models, and in our case are the same models used to transcribe the spoken keyword. There is a large amount of duplication of effort in the two passes, and the key to doing a fast search is to do the filler-only pass (half of the computation) and as much as possible of the keyword-and-filler pass ahead of time. This must not be at the expense of a large amount of storage as in the lattice-based approach of [16]. In [17], Knill and Young investigate various techniques for this, the two most effective being:

- *Approximated keyword recognition pass.* After the filler-only pass, the resulting phoneme boundaries and Viterbi scores at each boundary are stored. The keyword-and-filler pass performed at search time is then restricted to start and finish on a stored phoneme boundary rather than at arbitrary start and end frames. This simplification leads to a further 80% reduction in search time without affecting performance in any significant way.

- *Pattern Matching of Phoneme Strings.* A string-match can be performed between the keyword and the filler-only pass output extremely quickly, and whilst this is not an accurate enough process for doing the search, it can be used to eliminate a lot of the search space. Since both phoneme strings are output from a recogniser, the match must allow for recognition errors. This is done by having penalties for substitution, deletion and insertion, and using a Dynamic Programming (DP) algorithm to perform the match. Preprocessing in this way eliminates 85% of the search space with only a small drop in performance.

These two techniques give a 33 fold speed up, so the basic wordspotter only needs to work a little faster than real time to allow a reasonable search.

In the case of a hybrid ANN/HMM recognition system such as Abbot [10], there is a further opportunity to speed up the search by pre-computing and storing the posterior phone probabilities for each frame. This leaves very little computation to be done at search time. On a PC with 500MHz dual-processor and 512MB RAM we found we could search for 10 keywords (e.g. the 10-best transcriptions of a spoken keyword) at 300 times real time. This was *without* making use of either of the two speed-up techniques mentioned above. Since there are 45 probabilities per 16ms frame (for UK English), some compression is needed to reduce the storage requirements, which even with each probability represented by just one byte would be 1Mbyte for every 6 minutes of speech. We were interested to see if this could be reduced to a level which required little extra storage compared to even the lowest rate speech coders, and so targeted just 10 bits per 45 probabilities, i.e. 625 bits/sec. We achieved this using vector quantisation and a log(1+100x) transform applied to the probabilities before computing the distance measure, resulting in an error rate within 10% of the unquantised baseline system [18].

5. Compression of Speech Data

5.1 Requirements

For speech-as-data to be viable on a small device, the speech must not take up a lot of memory. There are numerous speech coding schemes, many being published standards, optimised for telephony. These offer a high perceptual quality (i.e. speech must sound natural and like the talker), but can afford to be a little unintelligible as the hearer can always request a repeat. For this reason a restricted 3.4kHz bandwidth has been used for telephony from the early days of frequency-division multiplexing until now.

For speech-as-data the perceptual quality need not be high, but the speech must be as intelligibile as possible. Phone numbers and alphanumeric details must be understood without error otherwise the entire concept fails. At the same time a high degree of compression is needed. The military solution to a similar requirement (high intelligibility, low bit rate) is the LPC10 vocoder [19]. This is a speech coder that relies completely on encoding the parameters of a model of speech production. Advances in parametric coding

[20] have substantially improved the quality (perceptual and intelligibility) of the LPC vocoder, and it is now close to the quality needed for speech-as-data whilst keeping a very low bit rate.

5.2 A Wideband Low-Bit-Rate Speech Coder

In [21], a speech coder is described based on LPC vocoding that improves intelligibility by extending the bandwidth of the encoded speech to 8kHz. The input signal is split into two bands, and the 0-4kHz band is encoded with a 10^{th} order LPC model, and the 4-8kHz band is encoded with a 2^{nd} order LPC model. The parameters are put through predictors and the residual is encoded using variable-rate Rice coding. Variable-rate coding is a luxury that stored speech can benefit from, which transmitted speech cannot. For extra intelligibility, the frame length is reduced from 22.5ms to 16ms. This reduction in frame length does not increase the bit rate proportionately as the smaller frame size is more predictable. At 2.4kbit/sec, the high-band parameters take up about 500 bit/sec. The interesting result from the Dynamic Rhyme Test (DRT [22]) reported in [21] is that when comparing the wideband and narrowband versions with both optimised for 2.4kbit/sec operation, the wideband coder performs 2.4 points on the DRT scale better than the narrowband coder. This demonstrates that even at this very low bit rate, bandwidth is more useful than finer quantisation.

5.3 Recognition of Compressed Speech

Various studies have shown that recognition performance degrades when recognising from speech compressed below 16kbit/sec [23,24]. For very low bit rate coders the problem is even worse. The ANN/HMM recogniser keyword-searching system described in section 4.3 can bypass this problem by preprocessing the speech at record time and therefore before compression, and can store the resulting data in less than 1kbit/sec. But for a general HMM-based recogniser, record-time processing can only go as far as encoding the acoustic features, which still require at least 4kbit/sec storage [25,26,27]. These could be stored alongside the speech, more than doubling the storage requirements, but in fact the two representations (coded speech and acoustic features) are only slightly different forms of the same information. It should be possible to derive the acoustic features *from* the coded speech, without spending the time decoding and re-analysing the speech with the associated loss in quality of a 'tandem' encoding.

The process of deriving spectrum-based features from the LPC parameters stored in a speech coder and the recognition performance of such a system are described in [18]. In fact the performance is as good as one with features derived directly from the original speech. However, the two parameter sets are clearly different since the error rate doubles when training is performed with one parameter set and testing with the other. Frame size appears to be an issue though - matching the coder frame size to the recogniser's 16ms frame gives virtually half the error of matching the recogniser's frame size to the coder's 22.5ms frame (using LPC-derived features for training and testing in both cases). This result may well be influenced by the fact the recogniser is designed to operate at 16ms. But it does mean that you are unlikely to get optimal performance just by using the LPC parameters from a standard coder, as frame sizes are normally in the 20 to 30ms range.

The performance is dependent on the amount of quantisation. Using the quantisation scheme described in [21] at 2.4 kbit/sec increases the error rate from 5.5% to 7.6%. The error rate is reduced to 7.1% if the training is performed on unquantised parameters, even though the recognition has to be done on quantised parameters, since the quantisation affects the variance but not the means of the parameters. This increase in error can be eliminated with finer quantisation at the expense of using a higher bit rate than 2.4 kbit/sec.

5.4 Reconstruction of Speech from Acoustic Features

The logical alternative to deriving acoustic features from coded speech is to reconstruct speech from the acoustic features derived for speech recognition. This is the approach taken in [28]. By adding a voicing and pitch detector to the acoustic feature extraction, speech similar in quality to LPC vocoding can be reconstructed. This approach has the advantage that a standard speech recogniser can be used (no retraining needed), but the disadvantage that the speech quality is not quite as good and the bit-rate is higher.

6. Conclusion

In this paper we have described technologies that enable speech to be used effectively as a data type in small, portable devices by providing means for organisation, browsing, searching and compression of the speech.

If the speech can be broken down into short segments only a few seconds long, and associated with the fields of an application, it will be easy to find the information and quick to listen to it. This organisation can be done either manually, by allowing the user to enter speech into any field as an alternative to text, or automatically with simple speech recognition responding to keyword labels that the user inserts into the recordings. Whilst the manual approach is preferable, the use of keyword labels allows information to be entered on-the-move, with eyes and hands busy.

For browsing longer recordings that cannot be broken up into short segments, a 2D random-access interface should be used in preference to tape-recorder style controls. This allows easy skimming and replay, and the ability to mark important sections with a visual icon for future reference.

When a large vocabulary recogniser is available as when connected to a larger machine, its output can be used without the need for checking and correction by applying summarisation techniques based on confidence score and inverse word frequency. The text becomes a quick way of locating relevant spoken information and need never be corrected unless the information is needed in textual form.

The other way of locating information is through a keyword search, where the user speaks a word or phrase, which locates the speech they are looking for. This search is more effective the longer the keyword. Ironically, searches with spoken keywords perform less well than searches with the same keyword entered as text, as the spoken keyword cannot be used in the search directly, but has to be transcribed into phonemes first. Speaking the word twice helps. A big issue is search speed - something in the order of 100 times real time is required, within the limited capabilities of a small device. This can be achieved by a combination of preprocesing at record time and simplifications to the search space. If an ANN/HMM hybrid recogniser is used, the posterior phone probabilities can be precomputed and stored in under 1kbit/sec, allowing a very fast search indeed.

To avoid using up memory with a lot of stored speech, a good compression scheme is needed. It needs to have high intelligibility to allow names, numbers and addresses to be retrieved from the speech - it also needs to work well with speech recognisers. This can be achieved with wideband (8kHz) encoding, which requires only 500 bit/sec extra and even at rates as low as 2.4 kbit/sec gives greater intelligibility than narrowband encoding. Good recognition performance can be achieved by deriving the acoustic features directly from the LPC parameters of the speech coder, but for best performance the LPC frame rate needs to be reduced to 16ms.

These techniques together go a long way towards giving stored speech the characteristics of text. This enables speech to be used for information input without the need for checking and correcting speech-to-text output, and with very modest computational resources. As a result, the device can be used on-the-move with eyes and hands busy, and can be small, low power and low cost without compromising functionality.

References

1. Pearce, D. Enabling new speech driven services for mobile devices: An overview of the ETSI standards activities for distributed speech recognition front-ends. In: Proc American Voice I/O society (AVIOS), May 22-24, 2000, San-Jose CA, USA.
2. Stifleman L J, Arons B, Schmandt C et al. VoiceNotes: A Speech Interface for a Hand-Held Voice Notetaker. In: Proceedings of INTERCHI, April 1993, ACM Press, 1993, pp 179-186
3. Arons B. Interactively Skimming Recorded Speech. PhD Thesis, M.I.T., 1994
4. Haddock, N. Structuring Voice Records Using Keyword Labels. In: Proceedings of CHI '96, ACM Press, 1996
5. Ades S, Swinehart DC. Voice Annotation and Editing in a Workstation Environment. In: Proceedings of AVIOS '86, 1986
6. Loughran S, Haddock N, Tucker R. Voicemail Visualisation from Prototype to Plug In: Technology Transfer through Component Software, HP Labs Technical Report HPL-96-145 1996.
7. Tucker RCF, Collins MJ. Speech Segmentation. U.S. Patent 6,055,495. April 2000.
8. Whittaker S, Hirschberg J, Choi J, Hindle D, Pereira F, Singhal A. SCAN: designing and evaluating user interfaces to support retrieval from speech archives. In: Proceedings of SIGIR99 Conference on Research and Development in Information Retrieval.1999. pp 26-33.
9. Koumpis K, Renals S. The Role of Prosody in a Voicemail Summarization System. In: ISCA Workshop on Prosody in Speech Recognition and Understanding, Red Bank, NJ, USA, Oct. 2001.
10. Robinson AJ, Hochberg MM, Renals S. The Use of Recurrent Neural Networks in Continuous Speech Recognition. In: Automatic Speech and Speaker Recognition. Kluwer, 1995, Chapter 19
11. Williams G, Renals S. Confidence Measures for Hybrid HMM/ANN Speech Recognition. In: Proceedings of Eurospeech '97, 1997
12. Valenza R, Robinson T, Hickey M et al. Summarisation of Spoken Audio through Information Extraction. In: Proceedings of ESCA ETRW Workshop on Accessing Information in Spoken Audio, April 1999, pp 111-116
13. Jones GJF, Foote JT, Sparck Jones K et al. Video Mail Retrieval using Voice: Report on Keyword Definition and Data Collection. University of Cambridge Computer Laboratory, Tech. Report No. 335, May, 1994
14. NIST. The Road Rally Word-Spotting Corpora (RDRALLY1). NIST Speech Disc 6-1.1, September 1991.
15. Knill KM, Young SJ. Techniques for Automatically Transcribing Unknown Keywords for Open Keyword Set HMM-Based Word-Spotting. Cambridge University Eng. Dept. Technical Report CUED/F-INFENG/TR 230, Sept 1995
16. James DA, Young SJ. A Fast Lattice-Based Approach to Vocabulary Independent Wordspotting. In: Proceedings ICASSP'94, Adelaide 1994, IEEE.
17. Knill KM, Young SJ. Low-cost Implementation of Open Set Keyword Spotting. In: Computer Speech and language (1999) 13, 243-266
18. Tucker R, Robinson AJ, Christie J et al. Recognition-Compatible Speech Compression for Stored Speech. In: Proceedings of ESCA ETRW Workshop on Accessing Information in Spoken Audio, April 1999, pp 69-72
19. Tremain TE. The Government Standard linear predictive Coding Algorithm: LPC10. Speech Technology 1982, pp 40-49
20. McCree AV, Barnwell III TP. A Mixed Excitation LPC Vocoder Model for Low Bit Rate Speech Coding. IEEE Trans. Speech and Audio processing, July 1995, 3:242-250
21. Seymour CW, Robinson AJ. A Low-Bit-Rate Speech Coder using Adaptive Line Spectral Frequency Prediction. In: Proceedings of Eurospeech '97, 1997
22. Voiers WD. Diagnostic Evaluation of Speech Intelligibility. In: Speech Intelligibility and Speaker Recognition, Mones E Hawley, Ed., Dowden, Hutchinson & Ross, Inc., 1977, pp 374-387
23. Euler S, Zinke J. The Influence of Speech Coding Algorithms on Automatic Speech Recognition. In: Proceedings ICASSP'94, Adelaide, IEEE, 1994, I:621-624
24. Lilly BT, Paliwal KK. Effect of Speech Coders on Sp eech Recognition Performance. In: Proceedings ICSLP'96, pp 2344-2347
25. Ramaswany GN, Gopalakrishnan PS. Compression of Acoustic Features for Speech Recognition in Network Environments. In: Proceedings of ICASSP'98, IEEE, 1998 2:977-980
26. Digalakis V, Neumeyer LG, Perakakis M. Quantization of Cepstral Parameters for Speech recognition over the WWW. In: Proceedings of ICASSP'98, IEEE, 1998 2:989-992
27. Speech Processing, Transmission and Quality aspects (STQ); Distributed speech recognition; Front-end feature extraction algorithm; Compression algorithms. ETSI ES 201 108 version 1.1.2. April 2000.
28. Chazan D, Hoory R, Cohen G, Zibulski M. Speech Reconstruction from MEL Frequency Cepstral Coefficients and Pitch Frequency, In: IEEE ICASSP 2000, Vol 3, pp 1299-1302