

Automatic speech recognition for scientific purposes - webASR

Thomas Hain, Asmaa El Hannani, Stuart N. Wrigley, Vincent Wan

Dept. of Computer Science, University of Sheffield
Regent Court, 211 Portobello, Sheffield S1 4DP, United Kingdom
{t.hain, a.elhannani, s.wrigley, v.wan}@dcs.shef.ac.uk

Abstract

We present ‘webASR’, an online interface to our state-of-the-art automatic speech recognition (ASR) systems. It aims to provide the wider scientific research community with an interface to speech transcription for domains and applications where the generation of such transcripts was not previously feasible. The webASR interface allows the upload of audio files and, in turn, the download of automatically generated ASR transcripts. Depending upon the specification given for an audio file, the system will transcribe using an appropriate speech recogniser chosen from one of the many available, such as a NIST RT evaluation system. The transcripts will be available for download after processing.

Index Terms: speech recognition, meeting transcription, web interface

1. Introduction

Automatic speech recognition (ASR) is now part of many products and applications, in areas ranging from medical transcription to game control, from call centre dialogue systems to information retrieval. Many thousands of people worldwide are employed in companies — both large and small — that provide speech recognisers specifically tuned for a customer’s needs. This remains important since ASR systems still perform best when trained and tested for a specific domain. Flexible speech transcription software is available from a smaller group of companies and mostly targeting the professional dictation market. ASR for other domains is usually embedded in applications and not freely accessible. Furthermore, in areas where error rates are still high there is little commercial interest.

In the *Information Age*, natural language processing applications are thriving and, similar to speech processing, attention is increasingly focused on texts which were previously unavailable in machine readable form such as broadcast news and discussions. More recently processing of transcripts of small to large group meetings such as tutorials, lectures, committee meetings, court-room proceedings, etc. have also seen increased interest. Current research is normally based on well annotated corpora such as the AMI corpus [1]. However, for many databases, speech transcripts are not included. Furthermore, most groups interested in natural language research do not have access to research-grade speech recognition software. Similarly, in other fields such as sociological or educational research, an increased demand for transcripts can be observed.

In this paper we present an online interface to our state-of-the-art speech recognition systems [2]¹. It aims to provide the scientific research community with an interface to speech transcription for *all* domains and applications where the generation

Description	Tot	CMU	AMI	NIST	VT
Initial decode	37.4	47.7	29.3	33.8	38.4
Adapted	28.2	37.9	21.9	24.6	27.9
Best single output	25.4	34.5	20.4	21.1	25.3
Combined	24.9	33.9	19.8	20.9	24.7

Table 1: %WER results on individual headset mic. (IHM) data of the AMI 2007 system on the NIST RT’07 evaluation set.

of such transcripts was not previously feasible. The *webASR* interface allows users to upload audio files and receive, in this version, transcripts in the form of raw text with time boundaries for utterances. Depending upon the specification given for an audio file, the system will transcribe using an appropriate speech recogniser (or part there-of) chosen from one of the many available, such as a NIST RT evaluation system [3]. The transcripts are made available for download after processing.

In the following we briefly describe the AMI/AMIDA meeting transcription technologies that form the base of the *webASR* system – the first such system available online. In order to allow efficient processing and straight-forward adaptability to systems we make use of the Resource Optimisation Toolkit (ROTK) which is briefly outlined in section 3. A description of the interface is followed by an example study of using the system for a realistic situation.

2. The AMI Meeting Transcription System

Transcription of speech in meetings is a task that has been under close investigation since 2002 in the form of U.S. NIST RT evaluations² and the AMI/AMIDA group has participated in these evaluations since 2005. NIST distinguishes between processing of multiple distant (MDM) and individual headset (IHM) microphone recordings. The complete AMIDA meeting transcription system used in the NIST RT’07 evaluation operates in a total of 10 passes. The initial pass serves to obtain a rough transcript to provide input to adaptation with VTLN, SAT, and MLLR. Subsequent passes then generate bigram word lattices which are expanded using 4-gram language models (LMs) and rescored using models that are differently trained (for example, using only meeting data, or adapted models, or different configurations in the feature extraction). Since a detailed description of the system would go beyond of the scope of this paper the interested reader is referred to [3] for details and a description of acronyms. Table 1 shows details for various stages in the system, from the initial decoding with unadapted models to the output of the best branch in the system. The outputs of several branches can then be combined, yielding the lowest word error

¹The first public system is expected to be online in July 2008.

²<http://www.nist.gov/speech/tests>

Description	Total	Sub	Del	Ins
Initial	44.2	25.6	14.9	3.8
Adapted	38.9	18.5	16.8	3.5
Final	33.7	20.1	10.7	2.9
Final - Man, Segments	30.2	18.7	9.4	2.0

Table 2: Performance results on multiple distant microphone (MDM) data of the AMI 2007 system on the NIST RT'07 evaluation set in terms of overall %WER, %Substitutions(SUB), %Insertion(INS) and %Deletions(DEL).

rate (WER). Data in this test set are taken from four different corpora (CMU, AMI, NIST and VT). The substantial difference in performance between these data sets originates mainly from different microphone qualities, even though heavily accented speech plays a role.

Table 2 shows results on the same data using MDM input and a less complex system structure. One can observe that the difference in the initial pass between IHM and MDM recordings is 7 % WER absolute which remains until the final pass. Whereas the difference between the manual and automatic segmentation of data on IHM was found to give only 1.3 %, it can be observed that for MDM the difference is 3.5 %.

The results above were obtained with a system that was specifically trained on multiple meeting sources. The webASR system is a modified version of the first pass of the MDM system but with 4-gram lattice expansion and confusion network decoding added (fig. 1) and with several modules re-tuned for speed.

3. The Resource Optimisation Toolkit

As mentioned above, speech recognition systems use many processing passes. A ‘processing pass’ normally only refers to a decoding step. However, systems include many other processing steps that can be computationally expensive (e.g. adaptation transform estimation or posterior based feature extraction [3]). As the ASR system does not operate in ‘batch mode’, full use of parallel computing can be made. The webASR systems uses the Sun Grid Engine³ (SGE) to distribute speech recognition processes on a compute cluster.

Optimisation of a speech recognition system on a single processor core mainly addresses efficiency of the processes individually. However, on a compute grid the overall latency of complex tasks such as the webASR system is determined by many factors and optimal distribution of processor load. Since the duration of processes is data dependent a prior optimisation should be replaced by compute resource management through load balancing systems such as SGE. This however requires that it be possible to split the system into as large a number of sub-tasks as possible. To this end we have developed the Resource Optimisation Toolkit (ROTK) that requires the definition of processing modules. Each module has predefined input and output and is requested to assess and define its computational split. A module processing graph can then be written to describe the process dependencies and the input/output relationships.

The webASR system shown in Fig. 1 is translated automatically into the process graph depicted in Fig. 2. Several recognition passes are started, working on different parts of the audio. Thus, where processes complete more quickly the graph can be transgressed faster. The overall latency is determined by the

³<http://gridengine.sunsource.net>

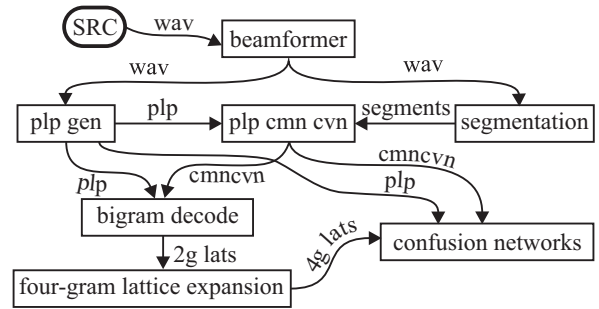


Figure 1: Module processing graph for webASR. Processing steps include beamforming, segmentation, computation of cepstral mean normalisation, bigram lattice generation and expansion to 4-gram lattices, followed by confusion network decoding. Each block represents a single process (or a predetermined number of identical processes) and the arrows indicate the data type being passed between modules.

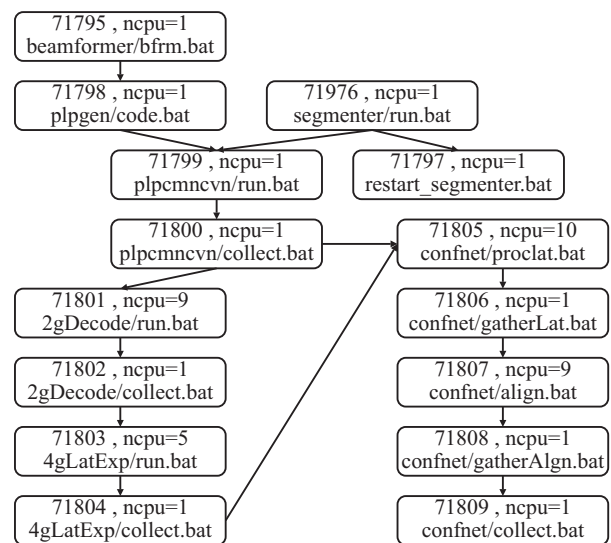


Figure 2: Processing graph associated with the module graph given in Fig. 1. Numbers denote process IDs and ncpu defines the number of identical sub-processes.

branches containing the slowest processes and thus need to start as soon as possible. ROTK scheduling and appropriate module definition ensure that this is achieved.

4. The webASR Interface

The interface to the ASR system is via a web-based application which runs entirely within the user’s browser. Access to the system is restricted to registered (and manually approved) users each of which are assigned a specific level of authority (e.g., administrator, public user, etc.) and upload quota (e.g., 1 GB over 30 days).

When logged in, the user has a number of options. The first is to edit their login and contact details to ensure that their profile is kept up to date on the system. The remaining options are concerned with the upload of audio files for recognition, management of uploaded files, and the download of the ASR transcript.

Registration New users register using a form on the sys-

tem homepage; required information includes the user’s name, email address (subsequently used as their username) and desired password. We also request that they provide a full postal address and telephone number. The registration request is logged by the system and an appropriate message is displayed when a member of the ‘administrator’ group logs in. The administrator reviews the registration request and, if suitable, approves it and assigns the user to a group. Although this step ensures tight controls on the user base, this process may become infeasible as the number of users increases and so approval may have to be automated in some manner or removed; this will be reviewed in the future. If approved, an email is automatically sent to the prospective user with an activation URL; once the user has clicked on this link, they have full access (commensurate with their account privileges) to the system.

File upload The process of uploading a file occurs in two stages. In the first stage, metadata about the audio file contents is collected which will be used in future versions to inform the recognition process. This metadata consists of the topics discussed, the number of speakers, the dialect and accent of each speaker; we also collect information regarding the specification of the microphones used to make the recording (e.g., lapel or far field, etc.).

The final stage involves the use of a digitally signed Java Applet to select and upload the file(s). Currently, the system is restricted to Microsoft WAV or NIST SPHERE formats sampled at 16 kHz and 16 bits; this restriction will be relaxed in future versions. The applet allows the user to browse their computer for appropriate files which, when selected, are displayed in a table. The applet is responsible for ensuring that the files are of the appropriate type and that the selected files would not exceed the user’s remaining quota.

File management and output retrieval The ‘My Account’ area (fig. 3) lists all the files that the user has uploaded. For each file, information regarding the original filename (the name used on the user’s computer), the file size and time of upload are provided. If any file upload failed (e.g., network congestion, manual abort) this is indicated and the user has the option of resuming the upload from the point of failure. Once the recognition process has finished for each file, a link to the transcript download is enabled.

Underlying architecture The web application used a standard design pattern known as the *model-view-controller* pattern (also known as the *front controller*, or *Model 2*) and was implemented entirely in Java.

A servlet (a means of mapping a URL to a special Java class) acts as the controller, providing a centralised point of control for all page requests. The controller is responsible for delegating requests to the appropriate Java class which processes the incoming request and data and makes any necessary changes to the underlying model. The model encapsulates all the domain-specific information on which the application operates. Upon successful completion, the controller then redirects the user’s browser to the new page. Here, the model (persistent storage mechanism) was implemented using a *MySQL* database and the view (the user interface / web pages) used *JavaServer Pages* (JSP).

5. Experimental Results

As the webASR system is based on the AMIDA system (Section 2), performance on meeting data is similar to that described in [3]. However, to the user of the webASR system it is of greater interest to show how the system will perform on data

Uploaded files (individually processed)








Filename	Size	Complete?		Upload timestamp	Transcript
REV-00002_x5mdm.wav	9.72 MB	yes	 	05-02-2008, 5:08 pm	
but.wav	87.00 KB	yes	 	06-02-2008, 11:00 am	
REV-00002_x5mdm.wav	9.72 MB	yes	 	06-02-2008, 11:54 am	
REV-00003_x5mdm.wav	85.78 MB	yes	 	06-02-2008, 4:49 pm	

Figure 3: Screen-shot from the webASR interface showing uploaded files and whether transcripts are ready.

that is substantially different to what the system is trained on.

Experiments were conducted using meeting recordings from an AMIDA project meeting (*amidaproj08*). Overall, 70 minutes were recorded and processed by the system as part of a demonstration of the capability of AMIDA technology. The meeting involved approximately 20 people (most of the training data for the system only had 4-5 participants per meeting) and was comprised of presentations by two speakers and extensive group discussion. The first language of most participants was not English. Data was recorded using a centrally located microphone array while presenters wore a lapel microphone. One of the main concerns in new domains is the mismatch of LMs; we now describe a priori attempts to improve LMs as well as experimental results.

5.1. Language models

A total of nearly 5 billion words of text were available for language modelling. These resources include various conversational telephone speech corpora such as Switchboard, Fisher, broadcast news corpora, and meetings corpora tailored for both conference room and lecture theatre style meetings. We also have collections of text downloaded from the internet for the meetings domain using techniques outlined in [4]. Notably, the same techniques can be used to adapt existing LMs by obtaining additional texts from the internet as part of the recognition. This works by using the initial recognition output to derive a text data collection from the internet from which a new LM can be built and interpolated into the existing LM.

Evaluation A 4-gram LM is built for each corpus using the SRILM Toolkit [5] with Kneser-Ney [6] discounting. These component LMs are then linearly interpolated to minimise the perplexity on a development text. Two development texts were used for this purpose depending upon the target domain of the language model: *rt06sconfeval* and *rt07slectdev* [3].

The RT07s conference room LM was created by interpolating the component LMs to minimise the perplexity on the *rt06sconfeval* development text. Likewise, the RT07s lecture theatre LM was optimised on *rt07slectdev*. For the experimental data *amidaproj08* no development data was available. However, due to the approximate topical area and the fact that both talks and discussions would be included, a mixture in equal proportions of the two development texts was used to estimate interpolation weights, further denoted as ‘mix’ LM. The weights of the most important components are shown in Table 3. As control, a fourth interpolated LM was built by interpolating directly on the hand transcriptions of *amidaproj08*, representing the best possible result. Note that the weights of the mix come close to this optimum.

Tables 4 and 5 illustrate the robustness of these LMs to changes in the test data. Table 4 shows the perplexity of each LM when tested on the two development texts and *amidaproj08*. The best LM in each case is the one optimised for that particu-

	conf	lect	mix	limit
Fisher	0.17	0.05	0.09	0.02
Fisher web	0.20	0.04	0.12	0.14
Hub5 lm96	0.03	0.05	0.05	0.06
AMI corpus	0.19	0.07	0.19	0.22
ICSI corpus	0.02	0.10	0.17	0.21
Meetings web	0.09	0.28	0.16	0.08
CHIL rt06train		0.06	0.01	0.02
CHIL preRT07data		0.10	0.04	0.01
RT06s conf web	0.05	0.11	0.04	0.04
RT06s lect web			0.06	0.03
AMIDA WIKI pages			0.02	0.08
AMIDA deliverables			0.01	0.07

Table 3: Interpolation weights of the various LMs. ‘conf’, ‘lect’ and ‘mix’ denote optimisation on rt06sconfeval, rt07slectdev and their combination respectively; ‘limit’ is the optimal LM based on amidaproj08. Only the most important components are included.

4g-LM	confdev	lectdev	testref
conf	73.1	140.8	142.5
lect	81.9	119.3	138.3
mix	78.3	130.5	133.8
limit	82.7	134.6	129.0

Table 4: Perplexities of various LMs across different test sets (confdev and lectdev denote RT’06 development sets while testref denotes amidaproj08).

lar text. Never-the-less, the LMs are fairly robust to a change of test set: the increases in perplexity are modest and the ‘mix’ LM seems to be more generic than others. Table 5 shows the corresponding WERs on amidaproj08 using the system depicted in Fig. 1. The improvements in WER with the ‘mix’ LM are due to the inclusion of meeting specific corpora (AMIDA deliverables and WIKI pages) and a lower OOV rate (Section 5.2). Note that the WER difference between the baseline system and the ‘limit’ is still relatively modest. The baseline represents the case where language models cannot be adjusted.

5.2. Vocabulary

The standard in the AMIDA systems is to use a 50000 word wordlist, generated by padding of in-domain lists [3]. For the ‘mix’ LM as described above a new wordlist was created to include the words from the AMIDA deliverables and WIKI pages. This can be seen as equivalent to providing documents about the topic of the meeting. A total of 54 new words were added and the overall Out-Of-Vocabulary (OOV) rate on the test data dropped from 2.53 % to 2.37 %. This suggests that the impact on WER would have been low.

LM	Reference PPL	lapel %WER	array %WER
conf	142.5	41.6	51.8
lect	138.3	41.6	51.2
mix	133.8	39.7	49.5
limit	129.0	39.4	49.4

Table 5: Perplexities and %WER of the four LMs for data recorded from the lapel microphone and the microphone array.

	lapel %WER	array %WER
webASR MDM first pass	39.7	49.5
RT07 MDM first pass	41.0	51.4
RT07 MDM final pass	35.3	43.6
RT07 IHM first pass	37.0	73.1
RT07 IHM final pass	31.3	72.2

Table 6: %WER on the amidproj08 using the webASR system and the AMIDA RT07 IHM and MDM systems.

5.3. Contrast Results

Table 6 shows WER results using the webASR system and the NIST RT07 systems, for close talking and far-field data. The difference between the RT07 systems and the webASR system lies in use of HDecode, lattice output and expansion to 4-gram lattices with the ‘mix’ LMs, and confusion network decoding. Thus the webASR achieves better first pass results. Additional passes in the RT systems allow a substantial reduction in WER, both on lapel and array recordings. The IHM system yields very good performance on lapel data even in the first pass. However, one can observe a considerable mismatch between IHM models and array recordings with almost twice the WERs. The MDM models appear to be far more robust and still yield reasonable performance on the lapel recordings and have thus been chosen to form the initial webASR configuration.

6. Conclusion

We have presented the webASR interface to the a state-of-the-art speech transcription system and have shown the relative robustness of the system. At the time of writing the system is still under development and is expected to go into trial use in July 2008 at <http://webasr.dcs.shef.ac.uk>.

7. Acknowledgements

The authors would like to thank all colleagues on the AMIDA project and especially its ASR group members without whose work this endeavour would not have been possible. We would also like to thank Adam Janin from ICSI for making the ICSI/SRI speech segmentation and clustering system available.

This work was partly supported by the European IST Programme Project AMIDA (Augmented Multi-party Interaction with Distance Access) FP6-033812.

8. References

- [1] J. Carletta et al., “The AMI meeting corpus,” in *Proc. MLMI’05*, Edinburgh, 2005.
- [2] T. Hain et al., “The AMI system for the transcription of speech in meetings,” in *Proc. ICASSP ’07*, 2007.
- [3] T. Hain, et al., “The 2007 AMI(DA) system for meeting transcription,” in *Proc. NIST RT07 Workshop*, 2007.
- [4] V. Wan and T. Hain, “Strategies for language model web-data collection,” in *Proc. ICASSP*, 2006, pp. 1069–1072.
- [5] A. Stolcke, “SRILM – an extensible language modeling toolkit,” in *Proc. ICSLP*, 2002.
- [6] R. Kneser and H. Ney, “Improved backing-off for m-gram language modeling,” in *Proc. ICASSP*, 1995, pp. 181–184.