A Distributed Approach to Speech Resource Collection

Raymond Molapo
Human Language Technologies
Research Group Meraka Institute
CSIR, South Africa
Multilingual Speech Technologies Group
North-West University
Vanderbijlpark
South Africa
Email: rmolapo@csir.co.za

Etienne Barnard
Multilingual Speech Technologies Group
North-West University
Vanderbijlpark
South Africa
Email: etienne.barnard@nwu.ac.za

Febe de Wet
Human Language Technologies
Research Group Meraka Institute
CSIR, South Africa
Email: fdewet@csir.co.za

***Abstract*—We describe the integration of several tools to enable the end-to-end development of an Automatic Speech Recognition system in a typical under-resourced language. Google App Engine is employed as the core environment for data verification, storage and distribution, and used in conjunction with existing tools for gathering text and for speech data recording. We analyse the data acquired by each of the tools and develop an ASR system in Shona, an important under-resourced language of Southern Africa. Although unexpected logistical problems complicated the process, we were able to collect a usable Shona speech corpus for the development of the first Automatic Speech Recognition system in that language.**

I. Introduction

The range of applications for high-quality automatic speech recognition (ASR) systems has grown dramatically with the advent of smart phones, in which speech recognition can greatly enhance the user experience. Currently, the languages with extensive ASR support on these devices are languages that have thousands of hours of transcribed speech data already collected. Developing a speech system for such a language is made simpler because extensive resources already exist. However for languages that are not as prominent, the process is more difficult. Many obstacles such as reliability and cost have hampered progress in this regard, and various separate tools for every stage of the development process have been introduced to overcome these difficulties.

The approach we explore in this paper is to combine these partial solutions. This process includes creating new tools and incorporating existing ones to develop an end-to-end ASR system in typical under-resourced conditions. The first stage of our solution uses an on-line tool called Rapid Language Adaptation Toolkit (RLAT) [1]. RLAT permits speech system developers to rapidly collect text data from the internet using web crawlers and web robots. We also incorporated an open-source software tool called Woefzela [2], that can collect speech data in resource-constrained environments at low costs. Google App Engine (GAE) [3] is the platform that houses the collected corpus in a reliable and secure location; tools were written to combine the outputs of RLAT and Woefzela for management, storage and distribtion via GAE. The end-to-end process uses a web interface to perform most of the tasks. The aim was to make the process as simple as possible, and facilitate ease of use.

II. Background

The foundation of most ASR and text-to-speech (TTS) systems is the availability of sufficient clean text and speech corpora. Most languages in developing and underdeveloped countries do not have the luxury of having such resources. Sixty percent of the world's population speak only about thirty of the 6900 living spoken languages, as native or second language speakers. The vast majority of the remaining languages are plagued by limited speech resources. Reasons for the lack of resources can range from native speakers being illiterate to accessibility because they live in remote areas. Languages that have little or no speech corpora are therefore classified as under resourced and those with sufficient speech corpora are said to be well resourced. The majority of African languages fall in the under resourced category, even though many of these languages have millions of speakers. For the current work, we focus our attention on the Shona language, which is a typical widely-spoken but poorly-resourced language in Southern Africa.

Fortunately, a substantial number of the under-resourced languages do have a significant presence on the World Wide Web. These internet sites can be crawled to retrieve the contents of the web pages, and the data can then be cleaned through suitable preprocessing stages to serve as general text corpora. The preprocessing steps include the removal of HTML tags, foreign-language content and various forms of punctuation.

For the specific purpose of ASR corpus development, suitable prompting material can be extracted from such general corpora. Woefzela employs short n-grams of frequently co-occurring words as prompts, in order to simplify the prompt-reading task. Thus, such segments need to be extracted from the text corpus. To avoid inappropriate or confusing prompts, it is useful to have the automatically-extracted segments verified by a native speaker before they are recorded. We have therefore developed a tool for on-line prompt verification, which integrates with Woefzela to download the selected prompts to Woefzela-enabled smart phones in preparation for speech recording.

The remainder of this paper is arranged as follows. Section III gives a brief background of the language/dialect Shona, chosen for baseline system evaluation. Section IV introduces the text data collection process and how the data were cleaned to generate prompts. Section V introduces the method we used to acquire speech data. Subsequently, Section VI reports on the experiments and results of each part of of the end-to-end system, and we conclude with some retrospective remarks on the strengths and weaknesses of the system that we have developed.

III. The Shona Language

The Shona language is a Bantu language native to the Shona people of Zimbabwe, southern Zambia, Botswana and parts of Mozambique. Shona is also used as an umbrella term to identify people who speak one of the Shona language dialects, namely Zezuru, Karanga, Manyika, Ndau and Korekore. Zezuru, mainly spoken in Mashonaland, is regarded as standard Shona dialect [4]. Shona is also spoken unofficially in South African and it is closely related to the Venda language (one of the official languages of South Africa). The language has more than 10.8 million first-language speakers across Southern Africa. Shona is a tonal language with two tones, high and low; the tones are not indicated in the script form of the language, which uses the Roman alphabet with a fairly regular relationship between orthography and pronunciation.

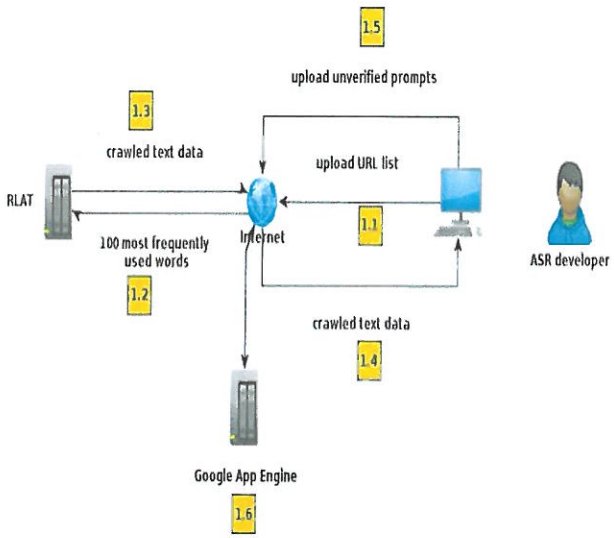
IV. Text Data Collection

There are various methods that can be utilized when collecting text data - see [5] for an overview. The method explored here was the use of RLAT, which is a tool developed at the Karlsruhe Institute of Technology to quickly collect data for a particular language without being a speech expert. RLAT can also be used for speech data collection, for the development of automatic speech recognition (ASR) and text-to-speech (TTS) systems. However, that functionality requires that audio data be recorded over the internet, which is often not feasible in developing countries, because internet connectivity is usually an non-existent. We therefore only utilize the text-collection capabilities of RLAT in our development.

A. Text Data Crawling

A list of one hundred frequently used Shona words, which was created by extracting text from a few Shona websites and performing a word frequency count, served as starting point for our development. The process flow of the remainder of the text data collection process is depicted in Figure 1. To collect text data, this list of one hundred frequently used Shona words was submitted to the RLAT team and used to search for web sites which may contain Shona content (based on the presence of those words). The Shona language option was added by the RLAT team to support our effort, and can be found on the drop-down menu on the RLAT website. For direct and robust web crawling, a text file with a list of eight URL's, shown in Table I, was also uploaded to the RLAT site. A total of 19 Megabytes of data was collected. The data contained approximately 267 thousand sentences, which include over 2.6 million word tokens. RLAT provides data clean-up mechanisms that remove HTML tags, punctuation marks and convert the text to lower case. This process is termed

Fig. 1. *A schematic diagram of the RLAT interaction process.*



language independent text normalization [6]. After the cleaning process, the data was found to have a large portion of English content: for both word types (i.e each unique words counted separately) and word tokens (i.e each word counted regardless of repetition) the ratio of English to Shona was approximately 1:1. Although some English data would be acceptable for our Shona development process, this ratio is too high - we therefore needed to perform additional processing, as described below.

TABLE I. *Shona URLs used to initiate crawling.*

Order	URL
1	http://mudararatinashemuchuri.blogspot.com
2	http://vashona.com/shona-news
3	http://www.watchtower.org/ca/jt/
4	http://www.kwayedza.co.zw/
5	http://www.voanews.com/shona
6	http://www.viva.org/downloads/pdf/wwp2012/
7	http://faraitose.wordpress.com
8	http://16dayscwgl.rutgers.edu

B. Prompt Design and Generation

There are several important factors that need to be kept in mind when designing prompts. These include the domain in which the prompts will be used, likely user populations and phonetic coverage of the prompts. The prompts were designed for open-domain purposes, which means a complete coverage is unlikely to be achieved (especially within the restricted scope of a corpus for an under-resourced language).

Since our prompts are intended for usage with Woefzela, we required short phrases that could easily be displayed on the screen of a smart phone. The developers of Woefzela found that prompts of three to five words work well for that purpose. Since Shona is a morphologically complex (agglutanitive) language with a conjunctive writing style, its words tend to

Fig. 2. *A screen shot of the on-line prompt verifier.*

Select Generated Prompts

☐	chichainda kumatunhu kunotaura
☐	chichakurukurwa muchitsauko chinotevera
☐	chichange chakatsva kuchapman
☐	chichasangana neesperance nesvondo
☐	chichatungamirirwa nakaputeni emmanuel
☐	chichava chiratidzo chokuvapo
☐	chichida kutsividza kukundwa
☐	chichisimudzira upfumi hwenyika
☐	chido chekupfuya hwai
☐	chief executive officer
☐	chief executive vesangano
☐	chifera na blessing
☐	chigaro chaka twasanuka
Project Name:	
Get Selected Prompts	Back to Main

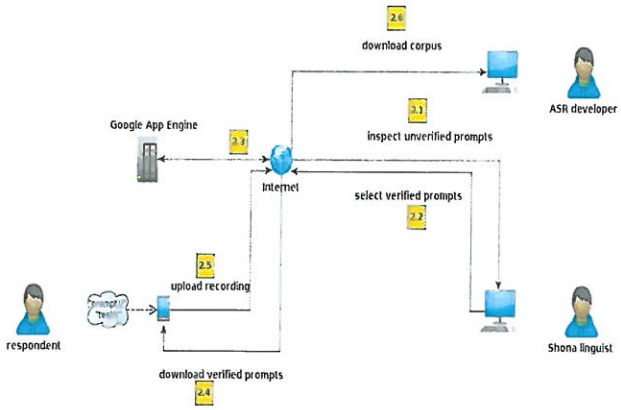
be long. Thus, the prompts were limited to 3-grams. This made the prompts not too long to read but still semantically meaningful. A prompt list of five thousand sentence fragments was generated. The digits in the text were not normalized to see how the native speakers would call out the numbers. The greedy algorithm used to generate prompts does not perform any spell checking. This can be overcome by providing a verification process - which is required in any case to remove inappropriate content, as we discuss next,

C. Prompt Verification

At this stage a text file containing five thousand sentences is generated for prompt verification. The UTF-8 encoded sentences have three words each for recording. Some of the sentences have mixed English and Shona words. Before the recording process could take place, the prompts had to be verified. This is to ensure that they do not contain spelling errors or inappropriate content (such as abusive or obscene phrases). The verification process was established through a web interface. The site was developed on Google App Engine in the django environment. The prompt text file can be uploaded by users to the site.

During the verification process, a text file is retrieved from the database and displayed in a table format. Each prompt has a corresponding check box which is checked if the verifier is satisfied with the correctness of the prompt. The interface provides a field that allows the user to give the file to be generated a unique name. The text file with verified prompts is saved to the database and is ready for download for use by the Woefzela application. Figure 2 shows a snapshot of the verification page.

Fig. 3. *A schematic diagram of the data collection and training processes.*



V. Speech Data Collection

The simplicity of the end-to-end process of ASR development lies in its process flow and automated nature. Figure 3 illustrates the interaction between different tools to accomplish end-to-end ASR system development. For the verified prompts to be recorded, they need to be downloaded from the Google App Engine server. This prompted the development of an application that facilitates the interconnection between the server and the recording tool. For this purpose, we developed an application called WDownload which retrieves the recently verified text file of prompts to the local mobile smart phone. It is open-source based and runs on the Android operating system.

A. Respondent Canvassing and Screening

For the recording process to start, native speakers of the language have to be recruited to perform verification and to do the recordings. A Shona native speaker was hired to be in every recording session to screen the respondents. The screening process was done by assessing the ability and fluency of how respondents could read fifteen Shona sentences that were randomly selected from the prompt text file.

The respondents included students and domestic workers, and were rewarded with token awards for their participation. However, this turned out to be surprisingly controversial – many potential respondents wished for substantial payments in order to participate, which was not compatible with the limited budget and open-source approach of the current project. Amongst the students, there was a greater receptivity for the open-source style; we were able to collect with greater success in that population, but only a limited number of students were available in Pretoria, where our collection was being performed.

Another challenge faced by field workers during the collection process was getting many respondents in a single location to record. This was the unexpected result of political events that had occurred previously. The recordings therefore had to be done with one or two respondents at a time in different

locations, and again limited our ability to collect a substantial number of speakers.

B. Respondent Registering

Respondents were required to sign a consent form to allow their voices to be used for our project; afterwards they received their tokens of appreciation. They were also required to fill in a profile field which included their age, phone numbers, identity numbers and their gender. The recording process using Woefzela (see below) was very intuitive for students: very little training was required to operate the application. The older generation needed more assistance on how the application should be operated.

C. Prompt Recording

Six inexpensive mobile telephones running the Android operating system were used to perform recordings. The phones had to be fully charged and running all the software required. Woefzela was used for audio and meta-data collection. Woefzela [2] is an open-source tool that runs on the Android operating system. It provides a practical and cost effective manner to collect speech data, especially in under-resourced environments.

Each respondent was required to record about 500 prompts initially; this was later reduced to 300 when frequent respondent fatigue and loss of concentration was noticed. Depending on how fast the respondent could read prompts, the recording session could take between 45 minutes to an hour. The recordings with the associated meta data were then saved onto the SD card. The data collection effort initially aimed at recording 20 Shona speakers, based on performance against speaker-number results previously obtained [7].

Collected data may be copied directly from the SD cards to limit reliance on the internet (a significant concern in the developing world). However, phones can be moved to an area with internet and directly upload all the files on the SD card to the server. WUpload is an Android application that is responsible for data upload to the Google App Engine. To ensure that the files are not duplicated, a checksum is returned from the server and if it matches that on the phone, the file in the SD card is deleted. The data is stored in a blob-oriented database for easy retrieval.

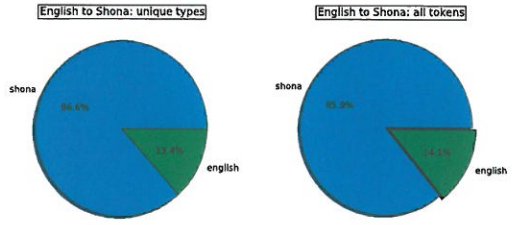
VI. Analysis, Experiments and Results

In order to evaluate the partial solutions that make up the end-to-end system, we have conducted various experiments as described in this section. We present the quality of the text data collected. We also analyse the recorded transcribed corpora, and finally report on the results obtained with an acoustic model that was trained to perform ASR.

A. Text Data Analysis

To control the amount of English text in our corpus, a list of English words was acquired by combining the CMU [8], Lwazi [9] and NCHLT English [2] pronunciation dictionaries. This list was used as a lookup table to remove sentences that contained English words only. The list consisted of 65 thousand words, mostly in the South African dialect of English. Sentences that had a mix of English and Shona were included in the corpus, since such code-switched speech is commonly found in ASR applications in under-resourced languages. Figure 4 shows pie charts that indicate the English-to-Shona ratio of the remaining sentences. Numerics were left as they are to hear how native speakers call them out (previously, we have found that numeric quantities are often pronounced in English [10]).

Fig. 4. *A schematic diagram of Shona to English text ratio.*



B. Speech Data Quality Control and Analysis

The speech data and associated eXtensible Mark-up Language (XML) files can be downloaded directly from the Google App Engine using a Python script. In order to complete the evaluation of our system, we have downloaded the data and developed a grapheme-based Shona ASR system.

Because of the complications described in Section V-A above, the collected corpus was smaller than we had initially intended: we had recordings from five female voices and six male voices, and a total of three and a half hours of speech. This data had to go through quality control measures both on the phone [11] and during off-line post processing [12]. The post processing scripts use the meta data from the mobile phone to tag the audio files. The process extracts the text prompts from XML files and creates associated transcriptions.

The off-line quality control examines the volume levels and the stop/start errors of the recording. Table II shows the results of the quality control process before any training is performed. From a total of 4018 recorded utterances, only 1855 were usable to train acoustic models. The recorded prompts had 7296 word tokens, containing 3619 unique types. The recording process managed to acquire 3.28 hours of speech data.

TABLE II. *Quality control data results.*

Respondent	Total Recordings	Usable Recordings
000	395	59
001	377	205
002	679	259
003	6	1
004	520	276
005	375	223
006	424	331
007	377	257
008	393	119
009	334	83
010	138	42
TOTAL	**4018**	**1855**

C. Recognition Results

The quality-control process verifies that prompts have appropriate durations and energy levels, but does not contain any mechanism to verify that the recorded audio files correspond to the prompted transcriptions. For a better understanding of the collected corpus, we randomly split the data into 80% and 20% training and test data respectively. For eleven speakers, we used five-fold cross validation. The speaker tags were generated so that no speaker would appear in both the test and training sets.

The recogniser employed standard Hidden Markov Model (HMM) based systems. For feature extraction, 39 dimensional Mel Frequency Cepstral Coefficient (MFCC) features were generated using HTK [13]. The MFCCs were extracted from a 25 milliseconds frame every 10 milliseconds. A flat grapheme-based language model was used for grapheme recognition. The dictionary used in the experiment was compiled from the crawled text data. It comprises a word list with the corresponding space separated grapheme representation. Table III shows the overall amount of data used and the accuracy of the grapheme-based system with both English + Shona and Shona-only data. The experiments were conducted using independent test sets and 5-fold cross validation. Table IV shows the results of the ASR system with both English + Shona and Shona-only data per speaker.

The training and test data contained English, which is a highly irregular language. The grapheme-based recognition results for such languages are invariably poor [14] – especially for the case where the majority of the speech data are written in the orthography of another language. To investigate this further, the English content from the training and test data were removed and the system was retrained. The last two columns of Table IV show the results for Shona-only training and test sets. It is observed that all the speaker results improved, showing that even the small amount of English data present in our corpus hurts grapheme-based performance substantially.

TABLE III. *Overall English + Shona and Shona-only results.*

Language	% Correct	% Accuracy	Amount of Data
English + Shona	66.29	55.34	3.28 hours
Shona-only	**73.95**	**64.68**	2.74 hours

TABLE IV. *English + Shona and Shona-only ASR results per speaker.*

	English + Shona		Shona Only	
Respondent	**% Correct**	**% Accuracy**	**% Correct**	**% Accuracy**
Speaker 000	60.08	48.34	61.64	52.05
Speaker 001	55.86	45.02	60.32	48.51
Speaker 002	64.84	53.48	69.03	58.07
Speaker 003	62.59	51.02	67.35	63.27
Speaker 004	70.39	60.44	73.28	64.83
Speaker 005	69.55	58.46	74.0	64.44
Speaker 006	67.73	55.64	69.40	58.58
Speaker 007	60.01	45.10	63.15	50.1
Speaker 008	**71.72**	**61.83**	**75.89**	**66.71**
Speaker 009	62.32	48.58	64.81	52.3
Speaker 010	65.11	48.19	67.19	55.96

Although our corpus was very limited in size and speaker variability, the grapheme accuracy achieved is acceptable. It is, for example, in the same range as the accuracies achieved for phoneme recognition on the 11 official South African languages during the Lwazi project [9]. Of course, this is only a starting point for Shona ASR development, and a number of measures that are likely to improve recognition accuracy are discussed below.

VII. Conclusion

We have explored the development of a set of tools that can be used for rapid end-to-end ASR system development. The process was tested and validated using the Shona language native to Zimbabwe. The system uses the web-based RLAT to acquire text data. The text data were cleaned to contain words in a 86% to 14% Shona-to-English ratio. Text data were segmented into prompts and uploaded to GAE. The prompts were verified on-line through a web-based system. To automate the end-to-end process, we also developed an Android application, WDownload, to download verified prompts to a mobile phone. Woefzela was used for recording and meta data collection. The recorded speech data was uploaded to the GAE through an Android application called WUpload. The data can be fetched at any time to develop an ASR system. With the combination of all these partial solutions, the end-to-end system development is made faster, easier, more intuitive and cost effective.

The accuracy of the ASR system can be improved in a number of ways. For example, a manually verified pronunciation dictionary - especially of the English words - would be useful. Also, during the recording process it was found that there were several inconsistencies in the pronunciation of certain numerals: the reading of years and large numbers, particularly, varied from respondent to respondent. This led to a degradation in word accuracy. Most importantly, more speech from a larger number of respondents will greatly enhance the accuracy of our recognizer. The logistical challenges that limited the size of our corpus were both unexpected and highly dependent on the local context. We hope that others will use our tools to perform ASR system development in under-resourced languages ... and that they will not be plagued by similar logistical issues!

Acknowledgment

The authors would like to thank Pedro Moreno for proposing the development of an end-to-end ASR collection toolkit. Tim Schlippe, Ngoc Thang Vu, Charl van Heerden, Nic de Vries, Neil Kleynhans, and the HLT Research Group, Meraka Institute, CSIR contributed to this project in various ways. Financial support from a Google Research Award is gratefully acknowledged.

References

[1] T. Schlippe, S. Ochs, and T. Schultz, "Wiktionary as a source for automatic pronunciation extraction." in *INTERSPEECH*, Makuhari, Japan, Sept 2010, pp. 2290–2293.

[2] N. J. De Vries, J. Badenhorst, M. H. Davel, E. Barnard, and A. De Waal, "Woefzela-an open-source platform for ASR data collection in the developing world," in *INTERSPEECH*, Florence, Italy, Aug 2011, pp. 3177–3180.

[3] D. Sanderson, *Programming Google app engine*. O'Reilly, 2010.

[4] C. Mudzingwa, "Shona morphophonemics: Repair strategies in Karanga and Zezuru," Ph.D. dissertation, University of British Columbia, 2010.

[5] A. Kivaisi and A. Mbogho, "Web-based corpus acquisition for Swahili language modelling," in *3rd workshop on Spoken Languages Technologies for Under-resourced languages*, 2012, pp. 42–47.

[6] T. Schlippe, C. Zhu, J. Gebhardt, and T. Schultz, "Text normalization based on statistical machine translation and internet user support." in *INTERSPEECH*. Makuhari, Japan: Citeseer, Sept 2010, pp. 1816–1819.

[7] E. Barnard, M. Davel, and C. Van Heerden, "ASR corpus design for resource-scarce languages," in *INTERSPEECH*, Brighton, UK, Sept 2009, pp. 2847–2850.

[8] R. Weide, "The CMU pronunciation dictionary, release 0.6." Carnegie Mellon University, 1998.

[9] Meraka-Institute. (2009) Lwazi ASR corpus. [Online]. Available: http://www.meraka.org.za/lwazi

[10] T. Ndwe, E. Barnard, and M. De Villiers, "Admixture practises in South African languages: Impact on speech-enabled technology design," in *IST-Africa Conference Proceedings*. IEEE, 2011, pp. 1–8.

[11] N. J. de Vries, M. H. Davel, J. Badenhorst, W. D. Basson, E. Barnard *et al.*, "A smartphone-based ASR data collection tool for under-resourced languages," *Speech Communication*, 2013.

[12] J. Badenhorst, A. De Waal, and F. De Wet, "Quality measurements for mobile data collection in the developing world," in *3rd workshop on Spoken Languages Technologies for Under-resourced languages*, 2012, pp. 139–146.

[13] S. Young, G. Evermann, D. Kershaw, G. Moore, J. Odell, D. Ollason, V. Valtchev, and P. Woodland, "The HTK book," *Cambridge University Engineering Department*, vol. 3, p. 175, 2002.

[14] W. D. Basson and M. H. Davel, "Comparing grapheme-based and phoneme-based speech recognition for Afrikaans," in *23rd Annual Symposium of the Pattern Recognition Association of South Africa*. PRASA 2012, 2012, pp. 144–148.