Continuous Sinhala Speech Recognizer

Thilini Nadungodage
Language Technology Research Laboratory,
University of Colombo
School of Computing, Sri Lanka.
hnd@ucsc.lk

Ruvan Weerasinghe
Language Technology Research Laboratory,
University of Colombo
School of Computing, Sri Lanka.
arw@ucsc.lk

Abstract

Automatic Speech Recognition has been successfully developed for many Western languages including English. Continuous, speaker independent speech recognition however has still not achieved high levels of accuracy owing to the variations in pronunciation between members of even a single community. This paper describes an effort to implement a speaker dependent continuous speech recognizer for a less resourced non-Latin language, namely, Sinhala. Using readily available open source tools, it shows that fairly accurate speaker dependent ASR systems for continuous speech can be built for newly digitized languages. The paper is expected to serve as a starting point for those interested in initiating projects in speech recognition for such 'new' languages from non-Latin linguistic traditions.

1 Introduction

Speech recognition has been a very active research area over the past few decades. Today, research on Speech Recognition has matured to a level where Automatic Speech Recognition (ASR) can be successfully implemented for many languages. Speech recognition systems are increasingly becoming popular because these systems create a friendly environment to the computer user. Speech recognition systems are able to provide an alternate and natural input method for computer use, particularly for the visually impaired. It could also potentially help to increase the overall productivity of general users by facilitating access programs and information more naturally and effectively.

In simple terms, speech recognition is the process of converting spoken words to machine-readable input. In more technical terms this can be stated as the process of converting an acoustic signal, captured by a microphone or a telephone, to a stream of words (Cole et al, 1996; Kemble, 2001).

Based on the two main types of human speech, speech recognition systems are generally classified into two types: discrete and continuous. In discrete speech, the spoken words are isolated. This means that in discrete speech, the speaker utters words in a way, which leaves a significant pause between words. Discrete speech recognition systems are created to recognize these isolated words, combination of words, or phrases and are referred to as Isolated (word) Speech Recognition (ISR) systems.

In continuous speech, the speaker pronounces words, phrases or sentences in a natural flow, so that successive words are dependent on each other as if they are linked together. There are no pauses or gaps between the spoken words in continuous speech. Continuous Speech Recognition (CSR) systems have developed to identify naturally flowing speech. The operation of a CSR system is more complex than an ISR system because they have to model dependencies between words.

Most of the speech recognition systems that have been developed so far are for English speech recognition. There is, however a lack of research in the field of recognizing non-Latin speech including many Indic languages and Sinhala. Sinhala is the mother tongue of majority of the Sri Lankans. It belongs to the Indo-Aryan branch of the Indo-European languages. Sinhala is also one of the official and national languages of Sri Lanka. Since there are many people in Sri Lanka who use Sinhala to communicate, there is

Conference on Human Language Technology for Development, Alexandria, Egypt, 2-5 May 2011.

a need to pay attention to the research area of recognizing Sinhala speech. When considering the existing domain of Sinhala speech recognition, almost all the researches that have been done so far are on discrete speech recognition. This is due to the difficulties of separating words in continuous speech and collecting sufficient sample data.

This paper presents the results of a research work carried out to recognize continuous Sinhala speech. The objective of this research was to apply existing continuous speech recognition mechanisms to develop a continuous Sinhala speech recognizer, which is not bound to any specific domain.

The rest of the paper is organized as follows. Section 2 overviews the works related to the speech recognition domain. Section 3 gives the design of the ASR system. Section 4 relates the implementation of the ASR. Section 5 presents the evaluation of the recognizer using error rates and live inputs. Finally Section 6 draws overall conclusion and describes possible future work.

2 Related Work

Interest in ASR steadily progressed from 1930s when a system model for speech analysis and synthesis was proposed by Homer Dudley of Bell Laboratories (Dudley et al, 1939). This system model was called the Vocoder. The intention of the originally developed Vocoder was to act as a speech coder for applications in the area of telecommunication, at that time. Vocoder was mainly involved in securing radio communication, where the voice has to be encrypted before transmission. As the time passed with the evolution of the technology, the Vocoder has also further developed and modern Vocoders are used in developing many applications for areas like linguistics, computational neuroscience, psychophysics and cochlear implant.

After Homer Dudley's Vocoder, several other efforts have been carried out in the area of designing systems for ASR. The early attempts of such research were mainly conducted based on the theory of acoustic-phonetics (Juang and Rabinar, 2005). Most of the early speech recognition researches were conducted by concentrating on recognizing discrete speech. In 1952, three researches at Bell Laboratories, Davis, Biddulph and Balashek built a speaker dependent system to recognize digits which were uttered as isolated words (Davis et al, 1952). Another discreet speech recognizer was developed by Oslon and Belar of RCA Laboratories. This system was a speaker dependent system and was capable of recognizing 10 syllables (Olson and Belar, 1956). In 1959 J.W. Forgie and C.D. Forgie at MIT Lincoln Lab built a speaker independent recognizer for recognize ten vowels (Forgie and Forgie, 1959).

In 1960s some Japanese laboratories proposed designs to build special hardware for the task of speech recognition. Among these the phoneme recognizer built by Sakai and Doshita at Kyoto University was the first to employ the use of a speech segmenter. This includes segmenting the input speech wave into several portions and analyzing each portion separately (Sakai and Doshita, 1962).

The idea of the Hidden Markov Model (HMM) first came out in late 1960s (Rabiner and Juang, 1986; Juang and Rabiner, 1991). An HMM was referred to as a probabilistic function set of a Markov chain. In 1980s at the Bell Laboratories, the theory of HMM was used to improve the recognition accuracy of recognizers which were used particularly for speaker independent, large vocabulary speech recognition tasks.

The concept of Artificial Neural Network (ANN) was reintroduced in late 1980s. A neural network is a software model which simulates the function of the human brain in pattern recognition. Early attempts of using ANNs for speech recognition were based on simple tasks such as recognizing a few words or phonemes. Although ANNs showed successful results with these simple tasks, in their original form they were found to be not suitable for handling complex speech recognition tasks.

In most speech recognition research up to 1980s, converting a speech waveform into separate words, which is the first step of the process of recognizer understanding human speech, was considered as a major problem.

As Juang and Rabinar (2005) shows, the researchers learned two important facts when the speech recognition field evolved. The first fact is that although the speech recognizers were developed using grammatical constraints in a language, the users mostly speak natural sentences which have no grammar and the inputs to these systems are often corrupted by various noise components. As response to this factor a keyword spotting method was introduced.

The second is that, as in human-to-human speech communications, speech applications often required a dialog between the user and the

machine to reach some desired state of understanding. Such a dialog often required such operations as query and confirmation, thus providing some allowance for speech recognition and understanding errors.

In late 1990s, real speech enabled applications were finally developed. Microsoft Windows XP and Windows Vista developed speech recognition systems for personal computers used in daily life (Microsoft). Also these applications were available not only for English language but also for many other languages. Although these ASR systems do not perform perfectly, they are already delivering real value to some customers.

3 Design of ASR

Building of an ASR system mainly consists of designing two models, namely the Acoustic Model and the Language Model. The Acoustic model is responsible for detecting phonemes which was spoken and the Language Model is responsible for detecting connections between words in a sentence. The following sections give the design of these two models.

3.1 Acoustic Model

An acoustic model is created using audio recordings of speech and their text scripts and compiling them into a statistical representation of sounds which make up words. This is done through modeling the HMMs. The process of acoustic modeling is shown in Figure 1.

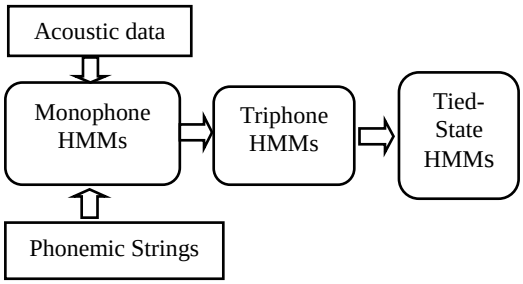

Figure 1. Block diagram of the acoustic modeling process.

3.2 Language Model

The way the words are connected to form sentences is modeled by the language model with the use of a pronunciation dictionary. The Language model of the proposed system is a statistical based language model as described in Rosenfeld (2000).

By assuming that the next word in the sequence will depend only upon one previous word, a bigram (2-gram) language model is created. Finally using this bigram language model a network which contains words in the training data is created. The process of language modeling is shown in Figure 2.

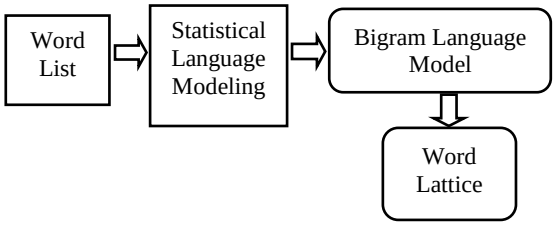

Figure 2. Block Diagram of the Language Modeling process.

3.3 Training

The basic procedure of building an ASR can be described as follows: the acoustic data of the training set goes through a feature extraction process and these features will be the input to the acoustic model training. The text transcription of the training data set is the input to build the language model. Trained acoustic model along with the language model is said to be the trained ASR system. The process of training the ASR is described in Figure 3. Next, the trained model goes through a testing process and the results obtained are used to adjust the trained model in order to get better and more accurate results.

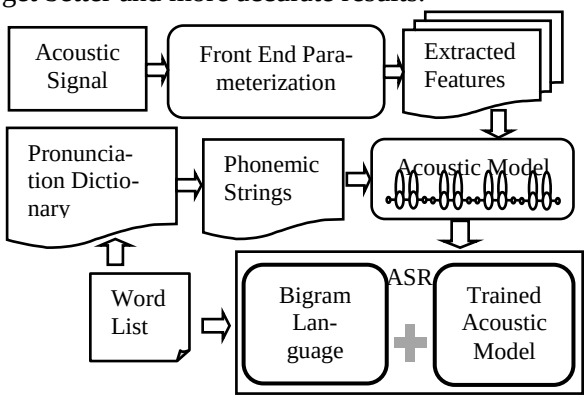

Figure 3. Block diagram of the process of training the ASR.

4 Implementation

This section describes the implementation of the ASR using Hidden Markov Model Toolkit (HTK) which was developed by the Cambridge University, UK (Young et al, 2006). HTK is primarily designed for speech recognition using Hidden Markov Models (HMMs). The steps of constructing the ASR can be divided a follows:

4.1 Data Collection

Unlike the English language, in Sinhala, written Sinhala and spoken Sinhala differ in some ways. While the grammar of written Sinhala depends on number, gender, person and tense, the grammar of spoken Sinhala does not follow them. Spoken Sinhala may vary in different geographical areas. So if we get the whole area of Sinhala language including both spoken and written Sinhala, we have to cover a huge vocabulary which is a very difficult and time consuming task. Hence, instead of covering the entire Sinhala vocabulary this research aims only the written Sinhala vocabulary, which can be used in automatic Sinhala dictation systems.

Before developing the ASR system, a speech corpus should be created and it includes recording continuous Sinhala speech samples. To build a better speech corpus the data should be recorded from various kinds of human voices. Age, gender, dialects and education should be the parameters which have to consider on collecting various voices. This requires a huge amount of effort and time. However this is the first attempt on building a continuous speech recognizer for Sinhala language and therefore as the initial step the data recording was done with a single female voice.

The first step in data collecting process is to prepare the prompt sheets. A prompt sheet is a list of all the sentences which need to be recorded. This sheet should be phonetically rich and cover almost all the phonetic transitions as possible. The prompt sheet was created using newspaper articles and the help of the UCSC 10M Sinhala Corpus. The prepared prompt sheet contained 983 distinct continuous Sinhala sentences for training purpose and these sentences were based on the most frequent words in Sinhala. The size of the vocabulary of data is 4055 words. For testing and evaluation purpose another 106 sentences were generated using the words contained in the previous set.

The prepared set of sentences was recorded using the software called *praa*t with the sample frequency of 16 kHz using a Mono channel. Each of the training utterances was recorded three times. The recorded files were saved in the *.wav format. The recording process was carried out in a quiet environment but they were not 100% without surrounding noise. This problem can be treated as negligible since both training and testing data are recorded in the same environment and therefore the noise will affect both data sets in an equal manner.

4.2 Data Preparation

The next step was to create the pronunciation dictionary. All the words used for the recordings are listed along with their phonetic representations in the pronunciation dictionary. Weerasinghe et al (2005) describes the phonemic inventory of the Sinhala language.

To train a set of HMMs every file of training data should have an associated phone level transcription. To make this task easier, it is better to create a word level transcription before creating the phone level transcription. The word level transcription was created by executing the Perl script prompts2mlf provided with the HTK toolkit using the previously prepared prompt sheet as input.

Using the created word level *MLF*, phone level transcription was created using the HTK label editor, *HLEd*. HLEd works by reading in a list of editing commands from an edit script file and then makes an edited copy of one or more label files. A HLEd edit script was used to insert the silence model 'sil' at the beginning and the end of each utterance.

Next, the features to be used in the recognition process should be extracted from the recorded speech signals. Feature extraction was performed using the *HCopy* tool. Here Mel Frequency Cepstral Coefficients (MFCC_D_A_E - 12 mel-cepstral coefficients, 12 delta coefficients, 12 acceleration coefficients, log energy, delta energy, and acceleration energy) was used to parameterize the speech signals into feature vectors with 39 features.

Building an ASR consists of creating two major models, Language model and the Acoustic model. The way the words are connected to form sentences is modeled by the language model and the acoustic model builds and trains the HMMs. In this project it creates a bi-gram language model. This was built using the HTK language modeling tools such as *LNewmap, LGPrep, LGCopy* and *LBuilt*.

By using the resulted bi-gram model a word lattice is created using the *HBuild* tool. This tool is used to convert input files that represent language models in a number of different formats and output a standard HTK lattice. The main purpose of HBuild is to allow the expansion of HTK multi-level lattices and the conversion of bigram language models into lattice format.

4.3 Training

The major process of building the ASR is building and training the acoustic model. The first step of this process was to create a prototype HMM. This prototype defines the structure and the overall form of the set of HMMs. Here a 3-state left-right topology was used to model the HMMs. The second step is to initialize the monophone HMMs. For this purpose HTK uses the *HCompV* tool. Inputs for this tool are the prototype HMM definition and the training data. HCompV reads the both inputs and outputs a new definition in which, every mean and covariance is equal to the global speech mean and covariance. So, every state of a monophone HMM gets the same global mean and covariance. Next a Master Macro File (MMF) called hmmdefs containing a copy for each of the required monophone HMMs is constructed. The next step is to re-estimate the stored monophones using the embedded re-estimation tool *HERest*. This process estimates the parameters of monophone HMMs from the training set that are intend to model. This is the process of training HMMs. The re-estimation procedure is repeated three times for each of the HMM to train.

After re-estimating the context independent monophone HMMs, we move onto context dependent triphone HMMs. These triphones are made simply by cloning the monophones and then re-estimating using triphone transcriptions.
The next step is to re-estimate the new triphone HMM set using the HERest tool. This is done in the same way as the monophone HMMs were estimated by replacing the monophone list and the monophone transcription with the corresponding triphone list and the triphone transcription. This process is also repeats three times.
The last step in the model building process is to tie states within triphone sets in order to share data and thus be able to make robust parameter estimates. However, the choice of which states to tie requires a bit more subtlety since the performance of the recognizer depends crucially on how accurate the state output distributions capture the statistics of the speech data. In this project it uses decision trees to tie the states within the triphone sets.

The final step of the acoustic modeling is the re-estimation of created tied state triphones and this process is also same as the earlier use of HERest. This is also repeated for three times and the final output is the trained acoustic model. Previously created language model is used to evaluate the trained acoustic model and the evaluation process will be described in the next section.

5 Testing & Evaluation

5.1 Performance Testing

As mentioned in the previous chapter in the data collection process, 106 distinct Sinhala utterances were recorded for the purpose of testing. Before starting the test, the acoustic data files and word level transcriptions were generated for the test speech set. Acoustic data files (files containing extracted speech features from the test set) were generated by executing the *HCopy* tool.

Next, the acoustic data of the test set was input to the built system and recognized using the Viterbi decoding algorithm. In HTK this is done by executing the *HVite* tool. The inputs to the HVite tool are, the trained acoustic model, coded acoustic data of the test set, language model word lattice and the pronunciation dictionary.

After generating these files the performance can be measured by comparing the manually created transcription file (file which containing the transcriptions of the input utterances) and the HVite generated output transcription file (file containing transcriptions of the recognized utterances). The computation of accuracy rate is done by executing the *HResults* tool. The above computation gives following results:

The percentage of 100% correctly identified sentences was 75.74% (i.e. 80 sentences out of 106 were perfectly correctly recognized). The percentage of correctly identified words in the whole set of test sentences was 96.14%. That is 797 words out of 829 words contained in the 106 sentences were correctly recognized.

5.2 Error Analysis

According to the above results an error analysis was done to identify the causes for the incorrect recognitions. When the incorrectly identified utterances were manually compared with the correct utterances, on most of the utterances only one or two syllables happened to be incorrectly identified. Very few utterances were incorrectly recognized due to incorrect word boundary detections. Only very few utterances were completely incorrectly recognized (as different words).

- Number of incorrectly identified utterances with one syllable changes = 17

- Number of incorrectly identified utterances due to incorrect word boundaries = 7
- Number of incorrectly identified utterances due to completely different words = 4

6 Conclusion

This paper describes an attempt to build an ASR system for continuous Sinhala speech. This section discusses the successfulness of the research, drawbacks and possible future works to improve the work carried out by this research.

6.1 Success & Drawbacks

The primary objective of this project was to build a prototype for a continuous Sinhala speech recognizer. As we were in a very early stage of building ASR systems for continuous speech, it can be said that the primary goal of using open source tools for building a recognizer for Sinhala speech has been achieved to a considerable and sufficient extent. The test results show that the system achieves 75% sentence recognition accuracy and 96% word recognition accuracy (or a word-error rate of just 4%). According to the error analysis it shows that most of the incorrectly identified utterances differed from the correct utterances only by one or two syllables. A better n-gram based language model could potentially help reduce such error further.

The system was trained only from a single female voice. Hence the above results were accurate only for the trained voice. The system gives a very low recognition rate for other human voices. This has to be solved by training the system using a variety of human voices of both male and female. Such an exercise is currently underway at the Language Technology Research Laboratory of the UCSC.

Another goal of this project was to build the system for an unrestricted vocabulary. The Sinhala language has a very large vocabulary in terms of its morphological and phonological productivity. We tried to achieve this goal by building the system using a sample of written Sinhala vocabulary. This vocabulary needs to be extended by adding words to the pronunciation dictionary and adjusting the language model according to it.

6.2 Future Work

The trained model can be improved to build a speaker independent speech recognition system by training the system using a large speech corpus representing voices from various kinds of human voices. To gain this target the speech corpus should consist of not only male and female human voices, but also should be representative in respect age group, education levels and regions.

Although speech recognition systems built for one language thus far cannot be used to recognize other languages, this research found that there is a large overlap between diverse languages at the phoneme level. Only a few phonemes of Sinhala differed from those of English. However, at the tri-phone level, the inter-dependence of phones with each other can be quite diverse between languages as well as different speakers of the same language. These features are being exploited by newer initiatives that have attempted to build 'universal' speech recognition systems.

Acknowledgments
Authors of this paper acknowledge the support of the members of the Language Technology Research Laboratory of the University of Colombo of School of Computing in conducting this research. Authors would also like to acknowledge the feedback given by two unknown reviewers who have helped in improving the quality of the paper. Any remaining shortcomings however are of the authors alone.

References

Cole, R. Ward, W. and ZUE, V. 1996. *Speech Recognition.* http://cslu.cse.ogi.edu/HLTsurvey/ch1node4.html.

Davis, K. H. Biddulph, R. and Balashek, S. 1952. *Automatic Recognition of Spoken Digits*. J. Acoust. Soc. Am. Vol.24, No.6. pp.627-642.

Dudley, H. Riesz, R. R. and Watkins, S. A. 1939. *A Synthetic Speaker*. Journal of the Franklin Institute. Vol.227. pp.739-764.

Forgie J. W. and Forgie, C. D. 1959. *Results Obtained from a Vowel Recognition Computer Program*. J. Acoust. Soc. Am. Vol.31, No.11, pp.1480-1489.

Juang, B.H. and Rabiner, L.R. 2005. *Automatic Speech Recognition – A Brief History of the Technology Development.* Elsevier Encyclopedia of Language and Linguistics.

Juang, B.H. and Rabiner, L.R. 1991. *Hidden Markov Models for Speech Recognition.* Technometrics. Vol. 33, No. 3, pp. 251-272.

Kemble, K. A. 2001. *An introduction to speech recognition*. Voice Systems Middleware Education. IBM Corporation.

Microsoft. *Windows Speech Recognition in Widows Vista*. http://www.microsoft.com/enable/products/windowsvista/speech.aspx.

Microsoft. *Speech Recognition with Windows XP* http://www.microsoft.com/windowsxp/using/setup/expert/moskowitz_02september23.mspx.

Olson, H. F. and Belar, H. 1956. *Phonetic Typewriter*. J. Acoust. Soc. Am. Vol.28, No.6, pp.1072-1081.

Pike, John. 2006. *Automatic Speech Recognition Techniques.* http://www.globalsecurity.org/intell/systems/asr-tech.htm .

Rabiner, L. and Juang, B. 1986. *An introduction to hidden Markov models*. IEEE ASSP Magazine, vol. 3, pp. 4-16.

Rosenfeld, R. 2000. *Two decades of statistical language modeling: Where do we go from here?*. Proceedings of the IEEE. vol. 88, pp. 1270–1278.

Sakai, J. and Doshita, S. 1962. The Phonetic Typewriter. Information Processing 1962. Proc. IFIP Congress, Munich.

Weerasinghe, A.R. Wasala, A. and Gamage, K. 2005. *A Rule Based Syllabification Algorithm for Sinhala*. Proceedings of 2nd International Joint Conference on Natural Language Processing, Jeju Island, Korea, pp. 438-449.

Young, S. Evermann, G. Gales, M. Hain, T. Kershaw, D. Liu, X. Moore, G. Odell, J. Ollason, D. Povey, D. Valtchev, V. and Woodland, P. 2006. *The HTK Book.* Cambridge University Engineering Department, pp. 1-14.