Incorporation of Syntax and Semantics to Improve the Performance of an Automatic Speech Recognizer.

by

Moyahabo Isaiah Rapholo

MINI-DISSERTATION

Submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Computer Science

in the

FACULITY OF SCIENCE AND AGRICULTURE

School of Mathematical and Computer Sciences

at the

UNIVERSITY OF LIMPOPO

Supervisor: Mr. MJD Manamela

Co-Supervisors: Prof HJ Oosthuizen

Dr N Gasela

2012

Declaration

I declare that the mini-dissertation hereby submitted to the University of Limpopo, for the degree of Master of Science in Computer Science has not been submitted by me for a degree at this or any other university; that it is my work in design and in execution, and that all material contained herein has been duly acknowledged.

_____ **22 March 2011**

Rapholo M.I. (Mr.)

[redacted]

Dedication

To my parents and siblings

Acknowledgements

I would like to thank my supervisor, MJD Manamela, for his feedback, support and patience throughout my Masters degree study. I will also like to thank my co-supervisors Prof HJ Oosthuizen and Dr N Gasela for their continued support. Lastly I would like to acknowledge University of Limpopo Telkom Centre of Excellence for Speech Technology, National Research Fund (NRF) and Telkom for their financial support for the study material that I required throughout.

Abstract

Automatic Speech Recognition (ASR) is a technology that allows a computer to identify spoken words and translate those spoken words into text. Speech recognition systems have started to be used in may application areas such as healthcare, automobile, e-commerce, military, and others. The use of these speech recognition systems is usually limited by their poor performance.

In this research we are looking at improving the performance of the baseline ASR systems by incorporating syntactic structures in grammar into an existing Northern Sotho ASR, based on hidden Markov models (HMMs). The syntactic structures will be applied to the vocabulary used within the healthcare application area domain. The Backus Naur Form (BNF) and the Extended Backus Naur Form (EBNF) was used to specify the grammar. The experimental results show the overall improvement to the baseline ASR System and hence give a basis for following this approach.

List of acronyms/ abbreviations

ABNF – Augmented Backus Naur Form

AM – Acoustic Modeling

ASR – Automatic Speech recognition

BNF – Backus Naur Form

DTW – Dynamic-Time Warping

EBNF – Extended Backus Naur Form

EM – Expectation Maximization

FBE – Flow-based Bayesian Estimation

FFT – Fast Fourier Transform

HMM – Hidden Markov Model

HTK – Hidden Markov Model Toolkit

IVR – Interactive Voice Response

LM – Language Modeling

LPC – Linear Predictive coding

MAP – Maximum A Posteriori

ME – Maximum Entropy

MFCC – Mel-Frequency Cepstral Coefficient

MIXBW-EM – Mixed-Bandwidth Expectation Maximization

MLF – Master Label File

MLLR – Maximum Likelihood Linear Regression

NB – Narrowband

NN – Neural Network

PLP – Perceptual Linear Predictive

PNCC – Power-Normalized Cepstral Coefficients

SNR – Signal-To –Noise Ratio

SRGS – Speech Recognition Grammar Specifications

SRI – Stress Repetitive Injuries

W3C – World Wide Web Consortium

WB – Wideband

WER – Word Error Rate

WSJ – Wall Street Journal

XML – Extensible Markup Language

List of Figures and Tables

Figures	**Description**	**Page**
Figure 1	Recognition process	8
Figure 2	Features of LM	22
Figure 3	Vector extraction	25
Figure 4	Task grammar	30
Figure 5	Adobe interface	33
Figure 6	Recognition process 2	42
Figure 7	Baseline results	45
Figure 8	Baseline Sketch	46
Figure 9	Language Models 1	47
Figure 10	Language Models 2	48
Figure 11	Syntax Structures 1	49
Figure 12	Syntax Structures 2	49

Tables

Table 1	Seltzer method	9
Table 2	Lopes method	10
Table 3	Baseline results	45
Table 4	Language models 1	47
Table 5	Language models 2	47
Table 6	Syntax results	49

Contents

Cover and Title i
Declaration ii
Dedication iii
Acknowledgements iv
Abstract v
List of Acronyms/ Abbreviations vi
List of Figures and Tables viii

1. Introduction 1
1.1 Background 1
1.2 Goals and Rationale 2
1.3 Motivation 2
1.4 Layout 3

2. Literature Review 5
2.1 Historical Background and current state 5
2.2 Similar Work 9
2.3 Types of ASR 12
2.4 Applications 13
2.5 Approaches to Design of ASR 15
2.6 Weaknesses and Flaws 17

3. Language and Acoustic Modeling and Grammar Specifications 18
3.1 Introduction 18
3.2 Grammar Specifications 18
3.3 Language Modeling 20
3.3.1 N-gram Language Models 22
3.3.2 Class N-gram Language Models 23
3.3.3 Sequence N-gram Language models 23
3.3.4 Maximum Entropy 23
3.3.5 Evaluating Metrics 24
3.3.5.1 WER 24
3.3.5.2 Test Set Perplexity 24
3.4 Acoustic Modeling 25
3.4.1 Speech Audio Characteristics 26

4. Speech Recognition Engine Development 27
4.1 Overview 27
4.2 Recognition Steps 27
4.3 Speech Data Statistics 28
4.3.1 Data Preparation 28
4.3.1.1 Task Grammar 29
4.3.1.2 The Dictionary 31
4.3.1.3 Recording Data 32
4.3.1.4 Creating Transcription Files 34
4.3.1.5 Phone-Level Transcriptions 35

4.3.1.6 Coding Data 36
4.3.1.7 Flat Start Monophones 37
4.3.1.8 Fixing Silence Models 39
4.3.1.9 Realigning Training Data 39
4.3.10 Triphones from Monophones 40
3.1.11 Tied State Triphones 41

4.4 Recognizer Evaluation 42
4.5 Conclusion 43

5. Experimental results and discussions 44
5.1 Introduction.....44
5.2 Baseline ASR System 45
5.3 Bigram and Trigram.....46
5.4 Baseline and syntax structures.....48

6. Conclusions.....50
6.1 Concluding remarks.....50
6.2 Recommendations.....51

References.....52

Appendix
A Test sentences 56
B Monophones.....60

Chapter 1

Introduction

1.1 Background about speech and speech recognition

Speech is the most natural means of communication between humans and it can be engaged without any “tools” or explicit education, hence the need to build and improve automatic speech recognition (ASR) systems in general. An ASR system is defined as the technology that allows users of computer-based information systems to speak entries rather than punching numbers on a keypad [1]. That is an ASR converts spoken words to text.

Speech recognition systems generally fall in the following main categories:

- **Speaker-dependent recognition system** - is developed to operate well for a single speaker. These systems are usually easier to develop, cheaper to buy and more accurate, but not very flexible
- **Speaker-independent recognition system** - is developed to operate well for any speaker. These systems are most difficult to develop, most expensive in cost and their accuracy is lower than speaker-dependent systems. However they are more flexible.
- **Speaker adaptive recognition system** - is developed to adapt its operation to the characteristics of new speakers. Its level off difficulty lies somewhere between speaker-independent and speaker-dependent systems.

Of course the size of vocabulary of a speech recognition system affects its complexity, processing requirements and accuracy.

A wide variety of techniques are used to perform speech recognition, there are many types of speech recognition systems and there are many levels of

speech recognition/analysis/understanding. Typically speech recognition starts with digital sampling of speech. The next stage is acoustic signal processing. Most techniques include spectral analysis; such as linear predictive coding (LPC), Mel-frequency Cepstral coefficient (MFCC), cochlea modeling and many more [2]. All these techniques mentioned above will be thoroughly explained in chapter two which covers literature review.

The next stage is recognition of phonemes, groups and words. This stage can be achieved by many processes such as dynamic-time warping (DTW), hidden Markov modeling (HMM), neural networks (NNs), expert systems and combinations of these techniques. These techniques will also be defined in chapter two. HMM-based systems are currently most commonly used and most successful approach in ASR.

1.2 Research goals and rationale

This research project aim to achieve the following goals:

- To enhance the design and development of an automatic speech recognition system focusing on restricted Northern Sotho vocabulary that can support a simple interactive voice response (IVR) health service application.
- The incorporation of syntax in the automatic speech recognition system will help enhance the training of such system to attain higher levels of accuracy and performance within its domain of application.
- To study a specific hidden Markov model toolkit (HTK), a toolkit that supports the development of automatic speech recognition systems based on HMMs.
- To develop a robust Northern Sotho ASR engine, using HTK that can be used for future projects.

- Train and test the automatic speech recognition system to see how much improvement, if any, is attained by comparing its performance with that of other similar systems.[1]

1.3 Motivation of the study

- After many years of speech technology research studies by institutions of higher learning and research centers world wide, the accuracy and performance of automatic speech recognition systems still requires much attention to reach acceptable levels among computer users of various spoken languages.
- People with disabilities can benefit from application of speech recognition technology. Speech recognition is especially useful to people who have difficulty in using their hands, ranging from mild repetitive stress injuries to involved injuries that preclude the using conventional computer input devices.
- Many electronic medical records applications can be more effective and may be performed more easily when deployed in conjunction with speech recognition system. Searches, queries and form filling may all be faster to perform by voice than using a traditional key board.
- We find that the challenge and efforts of designing automatic speech recognition and/or improving an existing ASR system within the confines of a restricted domain is likely to deliver improved results on the accuracy and performance of the baseline speech recognition system thus far developed.

[1] Where Northern Sotho is also known as Sepedi or Sesotho sa Leboa.

1.4 Layout of the research report

The mini-dissertation report is organized as follows: chapter two gives an in-depth analysis of literature study, in chapter three we look at speech recognition grammar specification, language modeling and acoustic modeling. The fourth chapter focuses on speech recognition engine development. In chapter five we look at the results and discussions. We make concluding remarks and suggest a few things for our future work in chapter six. Then lastly in chapter seven is our reference list.

Chapter 2

Literature Review

Automatic Speech Recognition (ASR) as defined previously in the first chapter (Introduction) is a computerized process that receives as its input a speech recording, and produces as its output an approximated transcription. In the computing world in general, speech recognition is a problem that is not completely solved; that is, no system has yet shown the ability to recognize speech as well as human can do it. This chapter looks at several approaches to improving ASR systems.

2.1 Historical background and current state of ASR

2.1.1 Historical background

Tremendous possibilities of ASR systems have been talked about for many years; beginning with the dawn of computer era over 4 decades ago. We have only recently started to see the usability of ASR systems over a wide-spread scale. An ASR system that would encompass real-time speech processing as well as language understanding is still considered to be many years away from us.

Initially, the research area of speech recognition was treated as a problem in statistical pattern recognition and its classification, using small vocabularies of isolated words or digits recorded in low-noise environments [1]. As technology developed, mathematical models were developed in the 1940's. The key invention of this era was the hidden Markov model (HMM) and the stochastic language model, which together enabled powerful new methods

for handling continuous speech recognition problem efficiently and with high accuracy in terms of performance [2]

2.1.2 Current state-of-the-art in ASR

The current research areas of automatic speech recognition include the following:

- **Language modeling** is any artificial language that can be used to express information or knowledge or systems in a structure that is defined by a consistent set of rules. The rules are used for interpretation of the meaning of components in the structure.
- **Adaptation to new speakers and environments.** When there is a mismatch between training and testing environments, speaker adaptation methods have been shown to be effective to reduce acoustic mismatches in ASR systems [1]. Among the typical adaptation methods such as maximum a posteriori (MAP), maximum likelihood linear regression (MLLR) and Eigen voice adaptation, MLLR is effective for limited adaptation data and can also reduce mismatch between training and testing environments by bias term as well as rotation term in a transformation matrix in model space. However, it exhibits severe performance degradation when the amount of adaptation data is extremely small. Eigen adaptation is more advantageous than the MAP and MLLR approaches when the amount of adaptation data is small.
- **Speaker, gender, and language identification.** Age and gender detection means: The intelligent speech recognition solution knows after a few words, whether the caller is male or female and from which age group he or she comes from, whether it was a best-ager, an adult, a young person or a child.

- **Hybrid recognition models.** Most current leading edge speech recognition systems are based on an approach called hidden Markov modeling (HMM). Traditional HMMs make some false assumptions, such as the fact that speech features occurring at one time are uncorrelated, and independent of other recently occurring features (even ten milliseconds earlier). SRI has developed a hybrid neural network/hidden Markov model speech recognizer that improves the accuracy of traditional HMM by modeling correlations among simultaneously occurring speech features and between current and recent features. More recent work involved modeling longer-term correlations and developing speaker adaptation approaches within this new framework [3].

- **Noise robustness.** Speech recognition systems work reasonably well in quiet conditions but work poorly under noisy conditions or distorted channels. For example, the accuracy of a speech recognition system may be acceptable if you call from the phone in your quiet office, yet its performance can be unacceptable if you try to use your cellular phone in a shopping mall. The researchers in the speech group are working on algorithms to improve the robustness of speech recognition system to high noise levels channel conditions not present in the training data used to build the recognizer.

The design of user-interfaces for speech-based applications is dominated by the underlying ASR technology. That is, the way interfaces are designed depends mainly on the kind of input speech data the system can handle. Figure1 shows the basic speech recognition process.

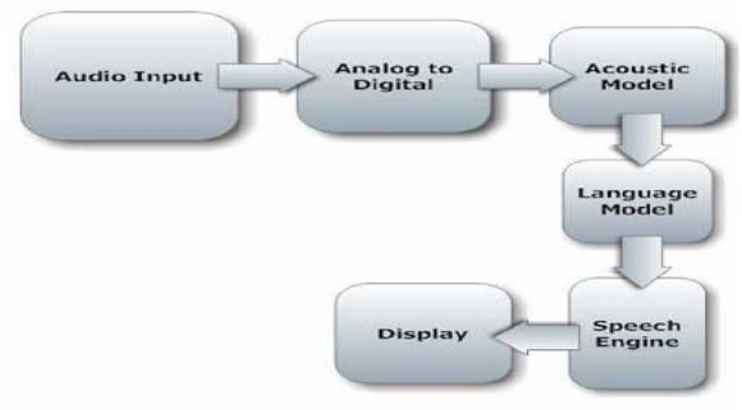


Figure1. The recognition process, adapted from Huang et.al [4]

Components of a recognition process

Audio input - with the help of microphone audio is input to the system, the PC sound card produces the equivalent digital representation.

Analog to digital - the process of converting the analog signal into digital form is known as digitization [4]. Digitization involves both sampling and quantization processes. Sampling is converting a continuous signal into discrete signal, while the process of approximating a continuous range of values is known as quantization.

Acoustic model - an acoustic model is created by taking audio recordings of speech, and their transcriptions, and using software to create statistical representations of the sounds that make up each word. Acoustic models break the words into the phonemes.

Language model - tries to capture the properties of a language to predict the next word in the speech sequence, compares the phonemes to words in its built dictionary.

Speech engine - the job of a speech recognition engine is to convert input audio into text; to do this it uses all sorts of data: software algorithms and statistical models

Display - this is where the recognized utterance is displayed.

2.2 Similar work

The performance of ASR technology has progressed to a point where commercial systems have been deployed successfully for some small to medium speech recognition tasks. The success of these systems has led to the desire for more widespread use of speech recognition technology. One serious difficulty in the deployment of ASR systems for new tasks is the expense of obtaining sufficient training speech data.

Seltzer et al [5] propose an alternative approach in which wideband acoustic models are trained using small amount of wideband speech and a large amount of narrow band speech. Seltzer et al also present a principled training algorithm based on expectation-maximization (EM) algorithm and show how this approach can be incorporated into the existing training schemes of HMM speech recognizers. Table 1 below shows the comparison of word error rate (WER) obtained using the flow-based Bayesian estimation (FBE) and the proposed mixed-bandwidth EM algorithm (MIXBW-EM) on the wall-street journal (WSJO) 20 000 test set for different proportions of wideband (WD) and narrowband (NB) training data.

Training Data	FBE	MIXBW-EM
20% WB + 80% NB	13.5	13.3
10% WB + 90% NB	18.3	13.9

Table 1 adapted from [5]

Judging from the results in Table 1 it is clear that the proposed method for training acoustic models for wideband speech recognition is an effective training method when collecting large amounts of wideband training data.

Rahman et al [8] introduces an Arabic phoneme recognition system using HMMs. Each Arabic phoneme is articulated, preceded by hamzah (the sign used in Arabic to represent the sound of a glottal stop) and ended by a silence (as described by Saibawaihi for studying the place of articulation of

phonemes) produces an ideal phoneme that is extracted and trained to obtain HMMs. These units are then used in the tests at the recognition stage. A new method is introduced for automatic segmentation of continuous speech to the level of the individual syllables. This method is based on classifying Arabic phonemes according to their level of perception. Results showed that the best feature to represent Arabic phonemes is weighted cepstral coefficients plus their differenced values in one observation vector (74%) while the area function is the worst features to represent Arabic phonemes (48%).

Lopes et al [8] propose an artificial intelligence approach to improving speech recognition. The method proposed here, is to analyze the output of any chosen speech recognition system, in order to determine whether a sentence contain syntactic or semantic errors. This is done via a software agent that uses the information from its knowledge base to attempt to correct the errors found. A system was implemented with a small vocabulary speaker-independent continuous speech recognition system, with limited sentence structures, the achieved increase in speech recognition accuracy, shows that there are benefits in using this approach. Table 2 below shows WER reduction for several tests.

	Test 1	Test 2	Test 3	Test 4	Test 5
Original WER	11.59%	11.34%	7.56%	22.75%	13.87%
WER after Agent	0.43%	0.84%	0.84%	8.15%	0.00%
WER reduction	96.30%	92.59%	88.89%	64.15%	100.00%

Table 2 adapted from [9]

Vizitiu et al [9] presents solutions for increasing environmental robustness of a Romanian language continuous speech recognizer, previously developed. The method is based on training the recognizer with degraded speech signal obtained by adding to speech various level of artificial noise. Experimental results show this scheme strongly increases the system robustness to additive noise.

Kim et al [10] presents a new feature extraction algorithm called Power-Normalized Cepstral Coefficients (PNCC) that is based on auditory processing. Major new features of PNCC processing include the use of power-law nonlinearity that replaces the traditional log nonlinearity used for MFCC coefficients, and a novel algorithm that suppresses the background excitation by estimating SNR based on the ratio of the arithmetic to geometric mean power, and subtracts the inferred background power. The results obtained show that the PNCC processing provides substantial improvements in recognition accuracy compared to MFCC and PLP processing for various types of additive noise. The computational cost of PNCC is slightly greater than that of conventional MFCC processing.

Ziolko et al [11] combines various topics in speech recognition. There are two central hypotheses. First one is that it would be useful to incorporate phoneme segmentation information in speech recognition and that this task can be achieved by applying discrete wavelet transform. The second main point is that adding semantics into language models for speech recognition improves recognition accuracy.

The research starts with analyzing differences between English and Polish from speech recognition point of view. English is a very typical positional language and Polish is highly inflective. Part of research is focused on aspects which should be changed due to the linguistic differences comparing to well known solutions for English to improve recognition of Polish. These

are mainly phoneme segmentation and semantic analysis. Phoneme statistics for Polish were gathered by the author and a toolkit designed for English was applied on Polish. The phoneme segmentation is more likely to be successful in Polish than English because phonemes are easier to be distinguished. A method based on the discrete wavelet transform was design and tested by the candidate.

Another part of research is focused on finding new ways of modeling a natural language. Semantic analysis is crucial for Polish because syntax models are not very effective and difficult to be trained due to non-positionality of Polish. This part of the thesis describes an unsuccessful approach of using part-of-speech taggers for language modeling in speech recognition and a much better bag-of-words model. The latter is inspired by well known latent semantic analysis. It is, however, easier to be trained and does not need calculations on big matrices. The difference is in the completely new approach to smoothing information in a word-topic matrix. Because of the morphological nature of the Polish language, this method gathers not only semantic content, but also some grammatical structure.

2.3 Types of ASR

Speech recognition systems can be divided into a number of classes based on their ability to recognize words or a list of words [12]. A few classes of speech recognition fall under the following but not limited to this categories:

2.3.1 Isolated speech recognition

Isolated words usually involve a pause between two utterances; it does not mean that it only accepts a single word but instead it requires one utterance at a time.

2.3.2 Continuous speech recognition

Continuous speech allows the user to speak almost naturally; it is also called the computer dictation.

2.3.3 Spontaneous speech recognition

At a basic level, it can be thought of as speech that is natural sounding and not rehearsed. An ASR system with spontaneous speech ability should be able to handle variety of natural speech features such as words run together and even stuttering.

2.3.4 Large vocabulary dictation

For repetitive stress injuries (RSI) and quadriplegics, and for formal document preparation in legal and medical services.

2.3.5 Interactive voice response (IVR)

For callers who do not have tone pads, for the automation of call centers and for access to information service such as stock market quotes [12].

2.4 Applications of ASR

There are new reports on the success of automatic speech recognition (ASR) technology for use with computers and the claims of how it will change the world. As an example of ASR technology, one can write a paper, compose an

e-mail message, and open programs without ever touching the keyboard. Here are some typical examples of the applications of ASR in the real world:

2.4.1 Medical perspective

People with disabilities can benefit from speech recognition programs. Speech recognition is especially useful for people who have difficulty using their hands, in such cases speech recognition programs are much more beneficial to the users [5]. Speech recognition is used in deaf telephony, such as voicemail to text. Speech recognition systems can also help in telemedicine. Telemedicine is defined as the use of telecommunication technologies such as the Internet to deliver medical information and services to locations at a distance from the care giver or educator.

This is part of the South African National Telemedicine Strategy that seeks to integrate the healthcare system by connecting and giving support to remote and rural medical centers of South Africa and most importantly, strengthening the referral systems.

2.4.2 Military perspective

Speech recognition programs are important in military, in Air Force speech recognition has the potential to reduce pilot work load and also such programs can be trained specifically to be used in helicopters and the airplanes [13].

2.4.3 Educational perspective

Individuals with learning disabilities who have problems with thought-to-paper communication can benefit from the software. Some students have a

problem with expressing their thoughts in writing but they can speak very well, so speech recognition software can be of great benefit to them [9].

2.5 Approaches to ASR design

Designing a machine that mimics human behavior, particularly the capability of speaking naturally and responding properly to spoken language, has intrigued engineers and scientists for centuries [6]. Since the 1930s, when Homer Dudley of Bell Laboratories proposed a system model for speech analysis and synthesis, the problem of automatic speech recognition has been approached progressively, from a simple machine that responds to a small set of sounds to a sophisticated system that responds to fluently spoken natural language and takes into account the varying statistics of the language in which the speech is produced.

2.5.1 Statistical based approaches

Variations in speech due to emotions, fatigue, stress, ill-health, accent and maturing from one stage to another (infant - adolescent, etc) are modeled statistically, using automatic statistical learning procedure, typically HMM. These approaches represent the current state-of-the-art in automatic speech recognition. This is due to their richness in mathematical structure which forms the basis for use. Another reason why HMMs are popular is because they can be trained automatically and are simple and computationally feasible to use.

2.5.2 Dynamic time-warping (DTW)-based speech recognition [2]

Dynamic-time warping is an approach that was historically used for speech recognition but has now largely been displaced by the more successful HMM-based approach. Dynamic-time warping is an algorithm for measuring similarity between two sequences which may vary in time or speed. For instance, similarities in walking patterns would be detected, even if in one video the person was walking slowly and if in another they were walking more quickly, or even if there were accelerations and decelerations during the course of one observation. DTW has been applied to video, audio, and graphics, any data which can be turned into a linear representation can be analyzed with DTW.

A well known application has been automatic speech recognition, to cope with different speaking speeds. In general, it is a method that allows a computer to find an optimal match between two given sequences (time-series) with certain restrictions; the sequences are “warped” non-linearly to match each other. This sequence alignment method is often used in the context of hidden Markov models.

2.5.3 Template-based approaches matching

“Unknown” speech is compared against a set of pre-recorded words in order to find the best match [14]. This has the advantage of using perfectly accurate word models. But it also has the disadvantage that pre-recorded templates are fixed, so variations in speech can only be modeled by using templates per word eventually becomes impractical. The basic idea is to align the utterance to each of the template words and then select the word or word sequence that contains the best [14].

2.6 Speech recognition weaknesses and flaws

Besides all these advantages and benefits, a “perfect” ASR system has not yet been developed. There are number of factors that can reduce the accuracy and performance of a speech recognition system [12]:

- Vocabulary size - in general increasing the vocabulary size decreases the recognition rate
- Number of speakers - with more than one speaker the ASR system must cope with the problem of speech variability
- Language complexity, gender, age, accent, environment conditions, all affect the performance of an ASR system

Chapter 3

Speech recognition grammar specifications (SRGS), language modeling (LM) and acoustic modeling (AM)

3.1 Introduction.

Automatic Speech Recognition (ASR) systems utilize statistical acoustic and language models to find the most probable word sequence when the speech signal is given [4]. Speech recognition engines usually require two basic components in order to recognize speech. One component is an acoustic model, created by taking audio recordings of speech and their transcriptions and then compiling them into a statistical representation of the sounds of words. The other component is called a language model, which gives probabilities of sequences of words. Hidden Markov Models (HMMs) are used as acoustic models and language model probabilities are approximated using n-grams where the probability of a word is conditioned on n-1 previous words [4].

3.2 Speech Recognition Grammar Specifications

Speech recognition grammar specification is a World Wide Web consortium (W3C) standard for how speech recognition grammars are specified. W3C is the main international standards organization for the World Wide Web [15]. A speech recognition grammar is a set of word patterns that the speech application will recognize. Automatic speech recognition systems that are speaker-independent, such as the LumenVox speech engine, are grammar based, meaning that they do not recognize any words that are not defined by the grammar [16]. Most grammars today are written according to the SRGS

format established by the W3C. SRGS includes two equivalent formats: a structured extensible markup language (XML) and a more human-readable Augmented Backus-Naur Form (ABNF) [17].

A common mistake that is often made is to try to develop grammars that cover every possible response a user could speak [18]. This attempt at comprehensive coverage introduces mere problems than it solves. In general, recognition accuracy is higher with smaller grammar. Larger grammars are more likely to include words or phrases that sound similar, and thus are likely to confuse the speech engine [18]. By keeping our grammar small, speech engines are able to better match what speakers say with words in the grammar. Even small alterations to grammars to account for very uncommon responses can be detrimental. A change to benefit a smaller percent of the users might negatively affect a larger percent of the users, causing a net loss in success. It is better to have simple grammar, and rely on the confidence score to tell you when the user has spoken outside the grammar [18]. We can then have a prompt explaining to the user that the system did not understand what was said. Most users will then re-phrase their utterance in a more predictable way.

Users who give wildly unexpected responses, such as those who swear at the system, should never be accommodated in grammars [18]. We should consider the responses as simply inappropriate input. It is always counter productive to try and deal with callers who are purposely misusing the application.

Here is an example of the Augmented BNF form of SRGS, as it could be used in a health care voice directory application:

```
// Default grammar language is Northern Sotho

// Single language attachment to an expansion
$Mothuši = (Mooki | Ngaka)!

// Handling language-specific pronunciations of the same word.
$people2 = Ngaka! Ya -mahlo | Ngaka! Ya- meno;

/**
 * Multi-function input possible
 * @example may I speak to the Doctor
 * @example may I speak to the Nurse
 */
Public $Kgopelo = ke kgopela go bona ($Mooki | $Ngaka);
```

Where | separate the alternatives, anything to the rite of = sign is called a production rule, /** */ represent comments that will not be processed by the compiler and $ is used to describe a noun word in simple terms.

3.3 Language modeling

Knowledge about languages is important in many fields including speech recognition and natural language understanding. The knowledge involves: lexical knowledge that is about the word definition and pronunciation, syntactic knowledge that describes the structure of a language, and semantic knowledge that shows the meaning of words and phrases [11]. LM, whose task is to estimate the possibility of occurrence of a sequence of words based on pre-acquired knowledge, is very critical to ASR [4].

In applications using language models, generally tasks can be described as finding the word strings most likely to match a sequence of observations [4]. The tasks are defined as:

$$\grave{\mathbf{W}} = \underset{w}{\text{argmax}}\, \mathbf{P(W \mid X)} = \underset{w}{\text{argmax}}\, \frac{\mathbf{P(W)\,P(X \mid W)}}{\mathbf{P(X)}} = \underset{w}{\text{argmax}}\, \mathbf{P(W)\,P(X|W)} \qquad \mathbf{(1)}$$

In the formula for this specific task, ***W*** is any possible word sequence, ***X*** is the acoustic observation, $\grave{\mathbf{W}}$ is estimated result. ASR problem can be finally decomposed to two parts, language modeling part ***P (W)*** and acoustic part ***P(X |W)*** corresponding to formula (1). For this section we focus on the language modeling part ***P (W).***

Statistical language modeling is the task of assigning a probability to a given sequence of words. For an ASR system, a good language model generally helps in two ways: 1) reduce search space for word sequence prediction and 2) improve recognition performance by providing context information [19].

For ASR system, the goal is to find the most likely sequence of words that match an utterance, and give the recognition result in text. The difficulty to this problem is that there may be a large number of candidate words for certain word positions [2]. For a speech recognition system involving a vocabulary of 30 000 words, the number of possible word sequences of a word string with 6 words is $(30\,000)^6$. This is a large space for searching, and in practice it is impossible to search. Language models provide useful probabilities estimation based on proper training test corpora can effectively reduce the search space by disregarding unlikely candidates. By looking at the following example it is easy to illustrate the above point:

Matome o nwa galase ya …..?/ Matome is drinking a glass of ….?

Words such as borotho (bread), Nama (meat), hlapi (fish), etc. are unlikely but words such as meetse (water), maswi (milk) are most likely. Language models help us with such choices. Figure 2 below shows basic features in language modeling

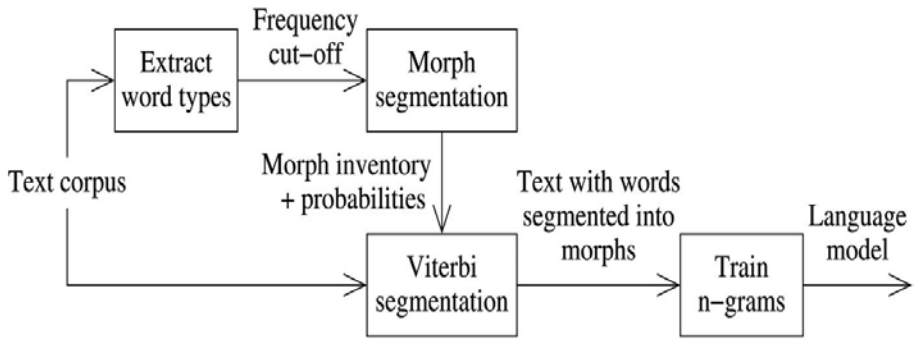


Figure 2: basic features of LM adapted from [19]

3.3.1 Word n-gram Language Models

The word n-gram language model is the most popular statistical language model in use. Given a certain word sequence, the probability of it occurring as a sentence is denoted as P (W), which can be expressed in the following way [3]:

$$P(W) = P(w_1, w_2, \ldots, w_n)$$
$$= P(w_1)\, P(w_2 | w_1)\, P(w_3 | w_2, w_1) \ldots P(w_n | w_1, w_2, w_{n-1}) \quad (2)$$

Using the chain rule and n-order Markov assumption, we assume a word appearance only depends on n previous words [4]. Therefore, if we define a language model under the assumption that the appearance of a possible word depends only on its previous two words, we get a trigram language model, with probability parameters estimated for a word group in the form of $P(w_i \mid w_{i-2}, w_{i-1})$. In the same way, if it is a bigram language model, the occurrence would be thought to rely on previous one word. The parameter is estimated by $P(w_i \mid w_{i-1})$. Although many different n-gram language models are used in various applications, the most widely used n-gram language models are the trigram language models.

3.3.2 Class N-gram language models

In order to cope with the data sparseness problem, class-based N-gram model was proposed. Instead of dealing with separated words, class-based N-gram estimates parameters for word classes. By clustering words into classes, a class-based N-gram model can reduce the model size significantly with the cost of slightly higher perplexity [4].

Suppose the probability of a word belonging to a certain class is $P(w_i \mid c_i)$, and the probability of a certain class following a history involving previous two words is $P(c_i \mid c_{i-2}, c_{i-1})$, giving a word that is uniquely mapped to only one class, as an example the trigram class computed based on previous two classes is formulated as :

$$P(w_i \mid c_{i-2}, c_{i-1}) = P(w_i \mid c_i)\, P(c_i \mid c_{i-2}, c_{i-1}) \qquad (3)$$

3.3.3 Sequence N-gram model

Sequence N-gram is an attempt to extend N-gram models with variable length sequences. A sequence can be a sequence of word, word class, part-of-speech or whatever a sequence of something that the modeler believes bearing important grammar information [19].

3.3.4 Maximum Entropy Language Model

Maximum Entropy (ME) model is an elegant and general statistical model that can incorporate features from different sources freely. A conditional ME model has the form:

$$P(w \mid h) = 1/z(h) \exp[\text{sum}(\text{lambda fi}(h, w))] \qquad (4)$$

Where lambdas are parameters, z (h) is a normalization factor and fi (h, w) are arbitrary functions of features of a word. A ME model that incorporates

trigram, distance N-gram, trigger pairs were observed more than 30% perplexity reduction over baseline trigram model [20].

3.3.5 Language Model Evaluating Metrics:

Primarily metrics are used to estimate the performance of language models in speech recognition systems. First, they are evaluated by the word error rate (WER) yielded when placed in a speech recognition system. Second, and more commonly, they are evaluated through their perplexity on test data, an information-theoretic assessment of their predictive power.

3.3.5.1 Word Error Rate (WER)

The most common metric for evaluating automatic speech recognition performance is WER, which is simply the ratio of the number of incorrectly recognized words in ASR output to the total number of words (N) in the reference text of the test set [4]. It is also used as a metric for language model performance evaluation. Recognition errors include insertions (I), deletions (D) and Substitutions (S). The WER is expressed as follows:

$$WER = \frac{D + I + S}{N} \times 100\ \% \qquad (5)$$

3.3.5.2 Test Set Perplexity

WER requires the evaluation of a speech recognition system, which is sometimes computationally costly. Another popular and simpler way for testing the performance of a language model is to measure the average probability assigned by a language model to each word given its linguistic context in the test set W. This measure related to cross-entropy is called test set perplexity [4].

$$PPL(W) = 2^{H(W) = 2^{(1/N) * \log 2\, P(W)}} = P(W)^{(-1/N)} \qquad (6)$$

3.4 Acoustic Modeling

Acoustic modeling of speech typically refers to the process of establishing statistical representations for the feature vector sequences computed from the speech waveform [21]. HMMs are one of the most common types of acoustic models. Other types of acoustic model algorithms include segmental models, super-segmental models, neural networks, maxim entropy language models and conditional random fields [21].

Acoustic modeling also includes pronunciation modeling which describes how a sequence or multi-sequences of fundamental speech units are used to represent larger speech units such as words or phrases which are the object of speech recognition [21]. Acoustic modeling may also include the use of feedback information from the recognizer to reshape feature vectors of speech achieving noise robustness in speech recognition. Figure 3 below gives a schematic representation of acoustic models with language models for recognition in a noisy environment

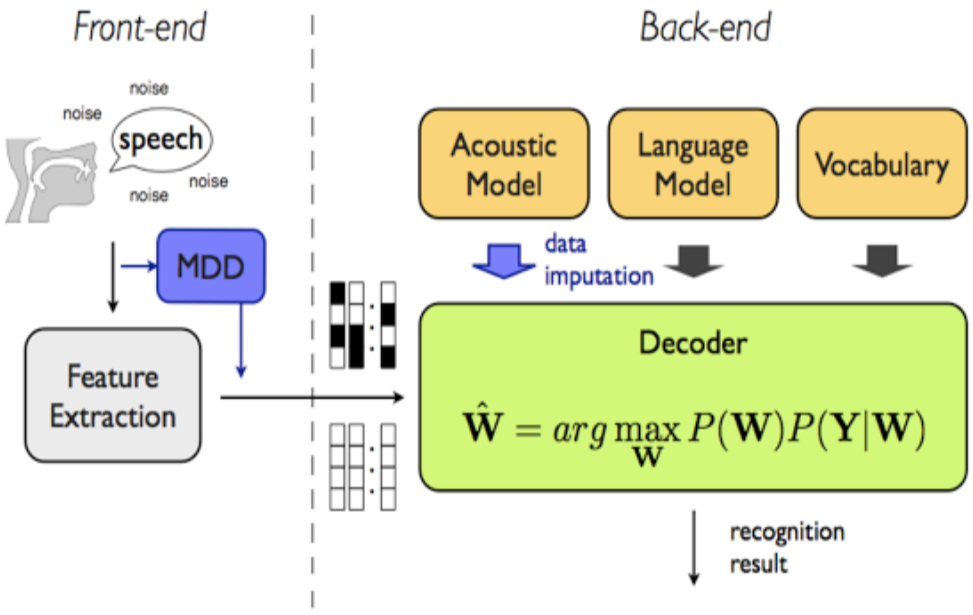


Figure 3. Diagrammatic representation of feature vector extraction for AM adapted from [22].

3.4.1 Speech Audio Characteristics

Different sampling rates can be used to encode speech data: 8 kHz, 16 kHz, 32 kHz, 44.1 kHz, 48 kHz and 96 kHz) are the most common ones, and different bits per sample (the most common being: 8-bits, 16-bits or 32-bits). Speech recognition engines work best if the acoustic model they use was trained with speech audio which was recorded at the same sampling rate/bits per sample as the speech being recognized [23]**.**

Chapter 4

Speech Recognition Engine Development

4.1 Overview of Speech Recognition Systems

Generally, ASR systems can be divided into two main categories: Pattern recognition systems compare input patterns to known stored patterns to determine a match and Acoustic Phonetic systems that use knowledge of the human body (speech production and perception) to compare features (such as vowel sounds) [5]. Most modern ASR systems focus on the pattern recognition approach because it interacts with recent ASR computing techniques and trends, and also it gives higher recognition accuracy.

4.2 Development of Speech Recognizers

1) *Audio recording and detection* – this can be achieved in many ways. Starting points can be found by comparing ambient audio levels with samples just recorded. Endpoint detection is a little difficult because speakers tend to leave "artifacts" including breathing, sighing, teeth chatters and echoes [9].
2) *Pre-Filtering* – is accomplished in a variety of ways, depending on the features of the recognition system. The most common methods are the "Bank-of-filters" which utilizes a series of audio filters to prepare sample, and the Linear Predictive Coding methods which uses a prediction function to calculate differences [21]. There are other forms of spectral analysis that are used in this regard.
3) *Framing\ Windowing* – it has to do with separating the sample data into specific sizes. The step also prepares the sample boundaries for analysis [21].

4) *Filtering* – additional filtering is not always present. It is the final preparation for each window before comparison and matching. This usually comprises of time alignment and normalization of sound waves [6].
5) *Comparison and Matching* – most of these techniques involve comparing the current window with known samples. These methods use HMMs, frequency analysis, differential analysis, linear algebra, spectral distortion, and time distortion methods [10].
6) *Action* (perform function associated with recognized pattern). This is where the recognizer performs the action as programmed by the developer, being recognizing an utterance or displaying the recognized utterance.

4.3 Speech Data Statistics

Audio is primarily an analog phenomenon. Recording a digital sample is done by converting the analog signal from the microphone to a digital signal through the A/D converter in the sound card. The important concept in audio recording is the "sample rate", which is how often to record the voltage values [23]. A good sample rate is about 8 KHz (8000 samples/sec) because speech is relatively low in bandwidth (mostly between 100HZ - 8 KHz).

4.3.1 Data preparation

The first stage of any speech recognition engine development is data preparation. Speech data is needed both for system training and testing. In the system that we built here, training speech data used was provided by the University of Limpopo Telkom Centre of Excellence for Speech Technology. We had to record the test data for this particular research project. The training data covers as much of the language as possible, Northern Sotho for this project, whilst the test data covers utterances that are likely to be uttered

within a specific domain, in our case the health care application area. The test data recorded consisted of 130 sentences with an average of six words per sentence whilst the training data consisted of 4000 sentences with an average of 7 words per sentence. The speech data that we recorded had to have good phonetic balance and coverage, so for that we needed to specify the task grammar.

4.3.1.1 Task Grammar

The task grammar defines constraints of what the speech recognition engine can expect as input. The grammar was defined in BNF and EBNF. Part of how the grammar was defined is listed hereafter:

```
$lefokwana = re a le thuxa|re go thuša bjang|naa re ka go thuša|molato ke eng|re go thuša ka eng;
$lsetho = leihlo|leoto|letlalo|letheka|letswele|lerapo;
$ditho = ditsebe|dimpa|dinoka|direthe|dikhuru;
$msetho = mala|mahlo|maoto|meno|maswafo|marapo|matswele;
$wasetho = molomo|mokokotlo|mogolo;
$yasetho = hlogo|nko|pelo|kgara;
$le = la ka le bohloko;
$m = a ka a bohloko;
$ya = ya ka e bohloko;
$wa = wa ka o bohloko;
(sent-start $lefokwana|($lsetho [$le])|($msetho [$m])|($wasetho [$wa])|($yasetho [$ya])sent-end)
```

where $lefokwana defines a phrase as anything between the equal (=) sign and semicolon (;), | stands for a logical or and sent-start, sent-end specifies the beginning and end of a sentence.

The HParse command with the above grammar definition produces a word network (*wdnet*) as shown in Figure 4.

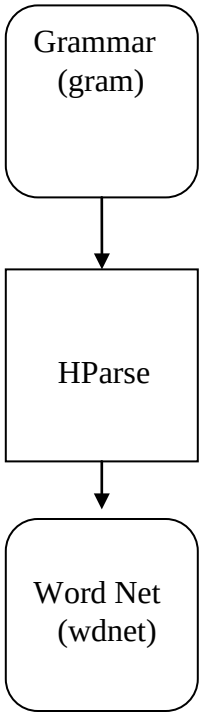


Figure 4. Task grammar adapted from [24]

The produced word network is as follows:

```
VERSION=1.0
N=37   L=67
I=0    W=!NULL
I=1    W=!NULL
I=2    W=SENT-START
I=3    W=re
I=4    W=!NULL
I=5    W=ka
I=6    W=go
I=7    W=thuxa
I=8    W=ka
I=9    W=eng
I=10   W=ke
I=11   W=a
```

```
I=12   W=lwala
I=13   W=o
I=14   W=kwa
I=15   W=bohloko
I=16   W=mo
I=17   W=kae
I=18   W=hlogong
I=19   W=ke
I=20   W=tla
I=21   W=go
I=22   W=fa
I=23   W=sengwenyana
I=24   W=ke
I=25   W=a
I=26   W=leboga
I=27   W=ke
I=28   W=tla
I=29   W=boa
I=30   W=gape
I=31   W=ge
I=32   W=ke
I=33   W=sa
I=34   W=be
I=35   W=kagone
I=36   W=SENT-END
```

4.3.1.2 The Dictionary

The first step in creating an "HTK-dictionary" is creating a sorted list of words (wlist) that is contained in the grammar definition. The "HTK-dictionary" gives words and their corresponding pronunciation. The wlist together with the *HDMan* command produces the dictionary. The following HTK command

HDMan –m –w wlist –n monophones1 –l dlog dict Dictionary

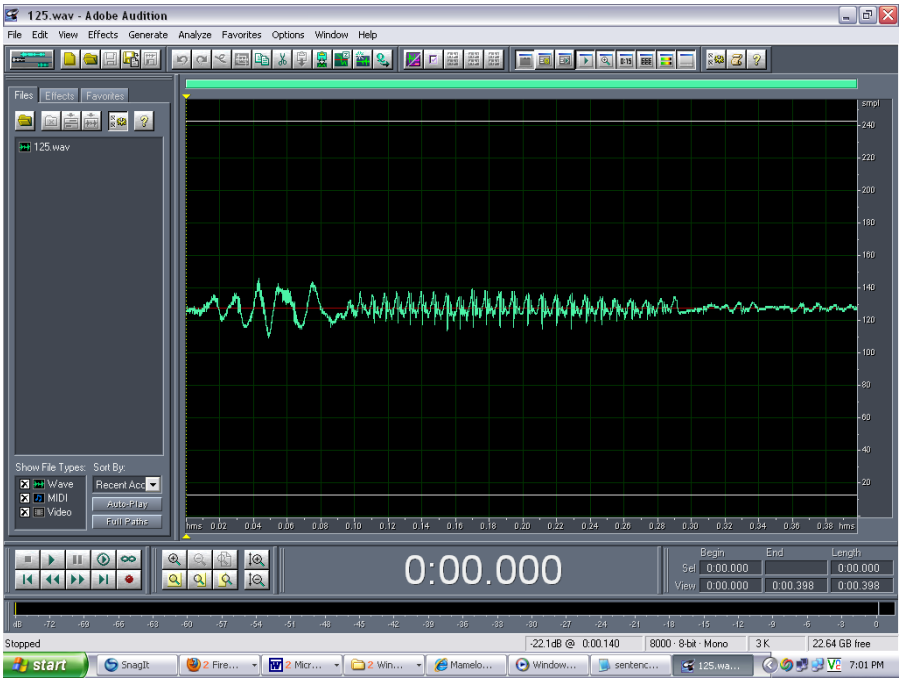
creates a new dictionary called dict from the old dictionary, where Dictionary is a Northern-Sotho pronunciation dictionary, and *dlog* is an output log file. The HDMan command is run with an edit script containing the following lines:

AS sp
RS cmu
MP sil sil sp

where "*sp*" is the short pause, "*sil*" the silence model and "*cmu*" refers to a style of stress marking in which lexical stress level is marked by a single digit appended to a word.

4.3.1.3 Recording the Data

As mentioned before, the training speech data used for this project was borrowed from the University Of Limpopo Computer Science Telkom Center Of Excellence. For the test data we recruited 15 volunteers within the University community. The room conditions where set in such a way that the dynamics of the external environments (noise in many forms) were minimized. Audio properties were set to a sampling rate of 8 KHz to be consistent with the training data. Adobe Audition version 1.0 was used in recordings as seen in Figure 5.



Figu**re 5. Adobe Audition Recording Software Interface adapted from [25]**

The command:

HSGen –n 130 -1 wdnet dict > sentacas.txt

Randomly generates 130 phrases from the dictionary and word network. Part of the phrases from the file “sentencas.txt” is as follows:

1. re go thuša bjang
2. mahlo a ka a bohloko
3. letlalo la ka le bohloko
4. nko
5. letswele
6. leoto
7. hlogo ya ka e bohloko
8. hlogo ya ka e bohloko

9. leihlo la ka le bohloko
10. mahlo
11. maswafo
12. mokokotlo wa ka o bohloko
13. matswele
14. matswele a ka a bohloko
15. nko ya ka e bohloko

4.3.1.4 Creating Transcription Files

Every file of training speech data must have an associated phone level transcription. So we create a single file that contains label entry for each line in the prompts file. This file is called the Master Label File (MLF). MLF can be created by running the Perl script ***prompts2mlf.***

A part of the MLF is shown below:

```
#!MLF!#
"*/su00101.lab"
lefeela
lefeela
tee
.
"*/su00102.lab"
madimetša
manamela
.
"*/su00104.lab"
monna
.
"*/su00105.lab"
masomenne
tee
.
"*/su00106.lab"
```

```
la
pele
jenewari
ngwaga
wa
sekete
makgolosenyane
masometshela
.
```

4.3.1.5 Phone-Level Transcriptions

The MLF file can further be subdivided into a phone level. The .mlf file together with *mkphones.led* under the command *HLEd* can help in that regard.

HLEd –d dict –i phones0.mlf mkphones.led sentecas.mlf

gives the following sample results: where each phrase is subdivided into its phonemes.

```
#!MLF!#
"su00101.lab"
sil
l
e
f
e
e
l
a
l
e
f
```

```
e
e
l
a
t
e
e
sil
.
```

The *mkphones.led* file contains the following lines:

```
EX
IS sil sil
DE sp
```

4.3.1.6 Coding the Data

In this section we want to extract speech acoustic feature vectors from raw waveform data. HTK supports both Fast Fourier Transform (FFT) and Linear Predictive Coding (LPC) based analysis, so we used Mel-Frequency Cepstral Coefficients (MFCC) [25]. But to do so we need a configuration file with the following line specifications and conversion parameters:

```
# Coding parameters
TARGETKIND = MFCC_0
TARGETRATE = 100000.0
SAVECOMPRESSED = T
SAVEWITHCRC = T
WINDOWSIZE = 250000.0
USEHAMMING = T
PREEMCOEF = 0.97
```

```
NUMCHANS = 26
CEPLIFTER = 22
NUMCEPS = 12
ENORMALISE = F
```

The configuration file together with the HTK tool HCopy converts audio wave files to MFCC. The MFCC representation captures all the necessary speech acoustic features extracted from the original speech waveforms.

4.3.1.7 Creating Flat Start Monophones

The first thing in HMM training is to define the prototype model, whose parameters are not relevant, so the important thing is the model topology definition. We create a file called "*proto*" with the following line specifications:

```
~o <VecSize> 39 <MFCC_0_D_A>
~h "proto"
<BeginHMM>
<NumStates> 5
<State> 2
<Mean> 39
0.0 0.0 0.0 ...
<Variance> 39
1.0 1.0 1.0 ...
<State> 3
<Mean> 39
0.0 0.0 0.0 ...
<Variance> 39
1.0 1.0 1.0 ...
<State> 4
```

```
<Mean> 39
0.0 0.0 0.0 ...
<Variance> 39
1.0 1.0 1.0 ...
<TransP> 5
0.0 1.0 0.0 0.0 0.0
0.0 0.6 0.4 0.0 0.0
0.0 0.0 0.6 0.4 0.0
0.0 0.0 0.0 0.7 0.3
0.0 0.0 0.0 0.0 0.0
<EndHMM>
```

Thereafter "*proto*" run with the script file "*train.scp*" under the command *HCompV*, with alterations "TARGETKIND = MFCC_0" to "TARGETKIND = MFCC_0_D_A" in the configuration file to create two files: "*vFloors*" and "*proto*". These two files were stored in the folder "*hmm0*". Another file called Macros was created by copying the contents of "vFloors" and combining them with "*proto*". A file called "*hmmdefs*" was created by copying each phone in monophones1 and adding the corresponding prototype in "*proto*".

We further created directories *hmm1, hmm2, …, hmm15* and executed the command line:

HERest –C config –I phones0.mlf –t 250 150 1000 –S train.scp –H hmm0/macros –H hmm0/hmmdefs –M hmm1 monophones0

for parameter estimations, where the numeric values, in the command line represent the pruning limits and can be varied depending on how well or poor the acoustic features of data are.

4.3.1.8 Fixing the Silence Models

In the previous step we created a three state left-to-right HMM for each phone for the silence model "*sil*". We then have to introduce extra transitions from states 2 to 4 and from states 4 to 2. This will make the model more robust as it allows individual states to absorb the various impulsive noises in the training data [25]. Furthermore, we create a one state "*sp*" model, which will have its emitting state tied to the center state of the silence model.
To achieve this first we created a file called "*sil.hed*" with the following line specifications:

```
AT 2 4 0.2 {sil.transP}
AT 4 2 0.2 {sil.transP}
AT 1 3 0.3 {sp.transP}
TI silist {sil.state [3], sp.state [2]}
```

The command:

HHEd –H hmm4/macros –H hmm4/hmmdefs –M hmm5 sil.hed monophones1.

is then executed. We further executed *HHEd* command twice for *hmm6* and *hmm7*.

4.3.1.9 Realigning the Training Data

The dictionary created has words with multiple pronunciations. So the phone models created so far can be used to realign the training data and to create transcriptions. This is done firstly by adding the silence model in the dictionary "*dict*". Then we run the HVite command from HTK. We also run HERest command twice for estimation of hmm8 and hmm9.

4.3.10 Creating Triphones from Monophones

Firstly, we needed to convert the monophones transcriptions to triphones transcriptions. Then we created a set of triphones models by copying the monophones models and re-estimating them. Lastly we tied states of triphones to ensure that all state distribution can be estimated better [25].
This was achieved by the following actions:
Create a file "*mktri.led*" containing the following line specifications:

```
WB sp
WB sil
TC
```

Then execute the command line:
HLEd –n triphones 1 –l (*) –I wintry.mlf mktri.led aligned.mlf
to align the training data.

We also ran the Perl script "***perl maketrihed monophones1 triphones1***" which makes triphones from monophones. A sample of triphones is listed below:

```
sil
l+e
l-e+f
e-f+e
f-e+e
e-e+l
e-l+a
l-a
sp
t+e
t-e+e
e-e
m+a
m-a+d
```

```
a-d+i
d-i+m
i-m+e
```

We then executed "*HERest*" twice and stored the results in *hmm10* and *hmm11*.

4.3.1.11 Making Tied-State Triphones

Now we have to tie states within triphone sets in order to share data and be able to make robust parameter estimation. The most important step in making tied-state triphones is to invoke *HDMan* against the entire Dictionary but not only "*dict*" to take into account phonetic balance and coverage. Then run "*HERest*" four times and store the results in *hmm12, hmm13, hmm14* and *hmm15*. This step is mainly to tie states within the triphone sets in order to allow sharing of data within the states and make robust parameter estimations.

4.4 Recognizer Evaluation

In this section we are to evaluate the performance of the speech recognizer. The recognition network and the dictionary have already been constructed, and test data has been recorded. The HTK recognition process is summarized in Figure 6 below.

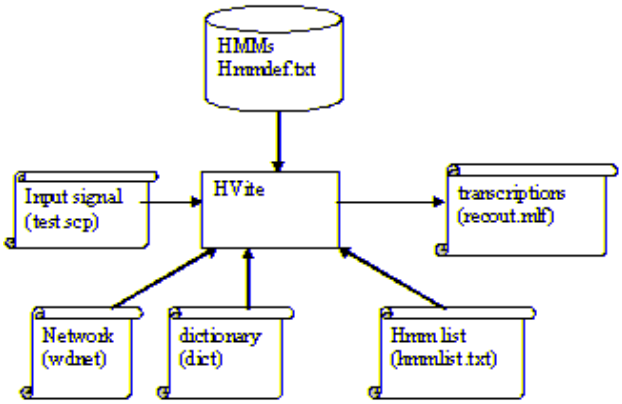


Figure 6. Recognition process 2, (adapted from [12])

To recognize the test data we execute the command line:

HVite –H hmm15/macros –H hmm15/hmmdefs –S test.scp –l '*' –i recout.mlf –w wdnet –p 0 –s 5 dict tiedlist

where test.scp contains a list of coded test files. Each test file will be recognized and its transcription output to an MLF file called recout.mlf, parts of which are displayed in the following output snippet:

```
#!MLF!#
"*/su00101.rec"
5800000 8200000 re -1906.449097
8200000 10600000 go -1886.889038
10600000 13900000 thuša -2566.635254
13900000 18600000 ka -3605.998535
18600000 30200000 eng -7362.350098
.


.

.

.
"*/su00221.rec"
3900000 8000000 molato -3142.889893
8000000 8600000 ke -532.481750
8600000 16000000 eng -4845.883789
.
```

4.5 Conclusion

This chapter described the pros and cons of the development of a speech recognition engine based on HTK toolkit. The results and discussions of results obtained at the end of the process will be discussed in detail in the next chapter.

Chapter 5

Experimental Results and Discussions

In this chapter we describe and discuss the results of HTK based experiments carried out in this research project. We also shown graphical details of the results obtained.

5.1 Introduction

The main objective of this research project was to add constraint domain specific syntax structures into an ASR restricted grammar and incorporate it to an existing Northern Sotho ASR engine, for purposes of improving the performance of a baseline ASR system. We also compare our results with the performance results of similar previous ASR projects, such as, the baseline ASR system enhanced with the bigram and trigram language models respectively.

The syntax structures were added using EBNF and BNF as described in chapter 4. We first state and discuss experimental results of an existing baseline ASR system. Then we describe the results of the baseline ASR that was enhanced with the bigram and trigram language models and lastly we will look at the baseline ASR that is enhanced with syntactic structures to its HTK grammar. The existing baseline ASR system will serve as a reference upon which improvements (if any) are measured.

5.2 Baseline ASR System

The experiment to test the recognition accuracy of the baseline ASR system was set in accordance with HTK specifications. The speech data waveforms were recorded at a frequency of 8 KHz and resolution of 8 bits. In this section, the grammar consisted of any sets of words, syntactically correct or not. The important condition was for the words to be from the same set (domain). The number of training speakers and testing speakers is 332 and 10 respectively.

The results in Table 3 and Figure 7 show that there was about 69% and 89% for both *sentence* and *word* recognition accuracy respectively. This Baseline ASR recognition results also confirms the baseline results part in Table 4 [26]. There are slight deviations due to the amount of training speech data used, but the training speech data was drawn from the same pool. We also give a schematic representation in terms of a bar graph in Figure 8.

5.2.1 The baseline ASR system results

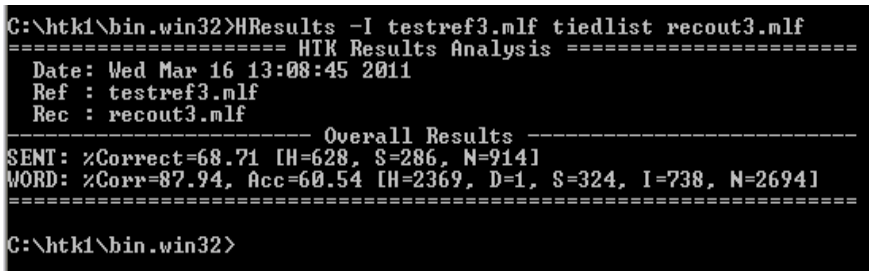


Figure 7. The baseline ASR results.

Accuracy	Percentage
Sentence	68.71 %
Word	87.94 %

Table 3. Baseline results.

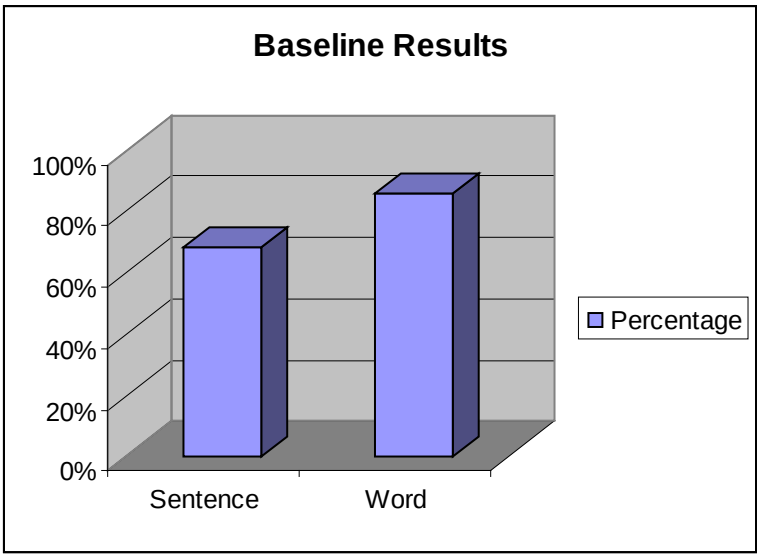


Figure 8. Graphical representation of the baseline ASR results.

5.3 Baseline ASR System enhanced with Bigram and Trigram Language Models

In this section we give an overview of the results obtained from [26], where the language models were incorporated into the baseline system for both training and new speakers.

The trigram language models bettered the bigram language model as well as the baseline system for both training and testing speakers in word and sentence recognition. Schematic representations for comparison are presented in Figure 9 and Figure 10 for training and testing speakers respectively. These results are mainly because of the language models' ability to predict "future" words based on "previous words". For large amounts of speech data trigram LMs work better than the bigram LMs as trigrams have more reference terms to use to predict the next phrase.

5.3.1 Results obtained with training speakers

Training speakers			
Accuracy	**Baseline ASR**	**Bigram**	**Trigram**
Sentence	51.1 %	75.8 %	78.9 %
Word	84.4 %	90.1 %	92.4 %

Table 4. The baseline ASR enhanced with LMs results (adapted from [26]).

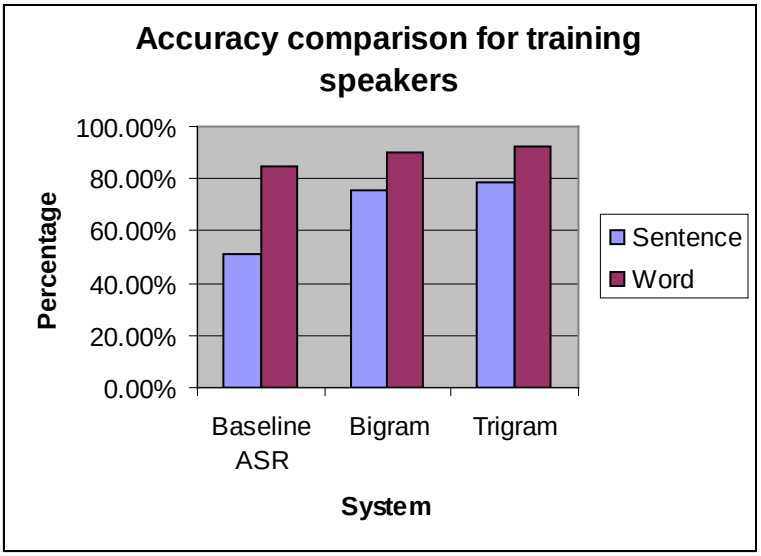


Figure 9. Language models for training speakers.

Testing Speakers			
Accuracy	**Baseline ASR**	**Bigram**	**Trigram**
Sentence	24 %	47.5 %	59.6 %
Word	63.2 %	80.4 %	84.4 %

Table 5. LMs for testing speakers (adapted from [26]).

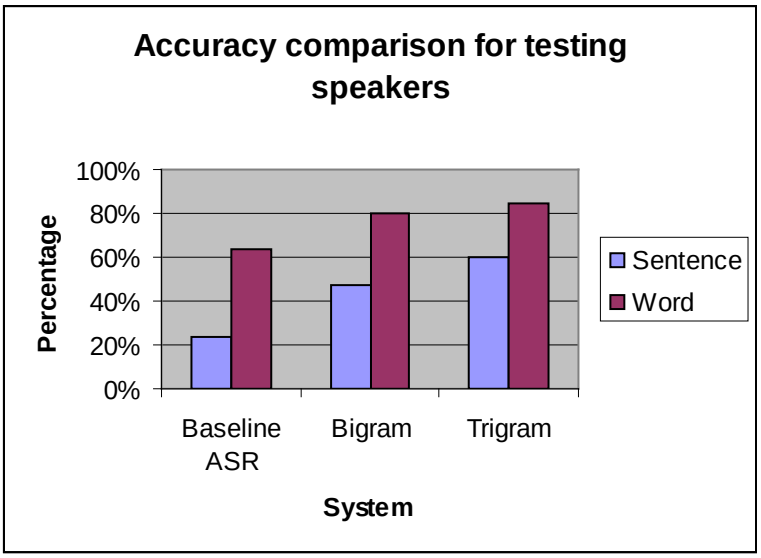


Figure 10. Language models for testing speakers.

5.4 Baseline ASR enhanced with syntactic structures

In this section we look at the experimental recognition results when the syntax for Northern Sotho grammar is incorporated into the ASR baseline system. The words are recognized as permitted by Northern Sotho grammatical structures.

5.4.1 Results for the syntax structures added to the baseline ASR

The overall recognition results show improvements when adding syntax structures to the HTK grammar. The recognition results in Figure 11 show 92.31 % and 93.81 % of both *sentence* and *word* recognition respectively. The results are also shown in Table 6 and represented schematically in Figure 12. These improvements are mainly because syntax structures reduce data sparseness. The results were improved further by down-sampling as the training data and test data were not prepared under the same environmental conditions and this speech data variability would negatively affect the results. If the sampling rate is too high the waveform “loses shape” and hence poor speech recognition results. That is some of the important features of the wave

are lost. Down-sampling “is the process of reducing the sampling rate of a signal” [5].

```
C:\htk1\bin.win32>HResults -I testref.mlf tiedlist recout.mlf
====================== HTK Results Analysis =======================
  Date: Wed Mar 16 15:57:49 2011
  Ref : testref.mlf
  Rec : recout.mlf
------------------------ Overall Results --------------------------
SENT: %Correct=92.31 [H=360, S=30, N=390]
WORD: %Corr=93.81, Acc=89.05 [H=591, D=9, S=30, I=30, N=630]
===================================================================

C:\htk1\bin.win32>
```

Figure 11. Syntax structures added to the baseline ASR.

Accuracy	**Percentage**
Sentence	92.31 %
Word	93.81 %

Table 6. Syntactic structures added to the baseline ASR.

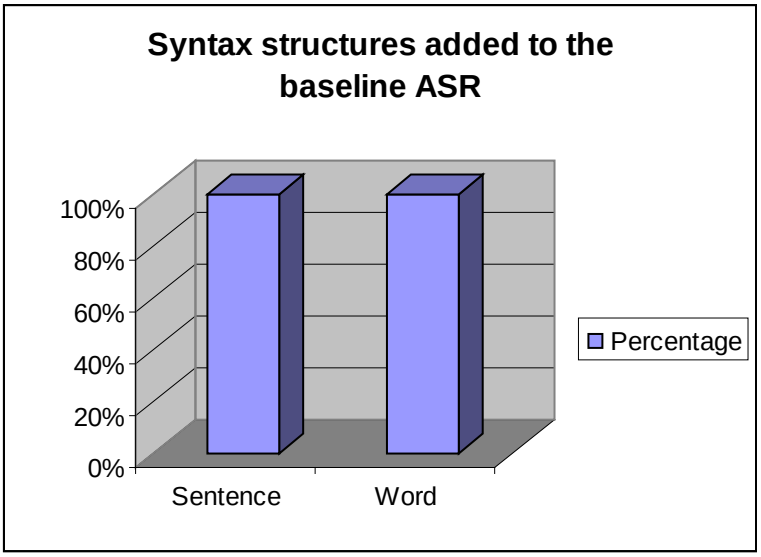


Figure 12. Graphical representation of syntax structures added to baseline ASR.

Chapter 6

Conclusions

6.1 Concluding remarks

The main objective of this research project was to develop and incorporate syntax structures into a Northern Sotho automatic speech recognizer to improve its recognition performance. Although the scope of this ASR research project was limited and constrained to a health-care domain, there appears to be enough evidence from the experimental results that this approach promises to yield improved accuracy results where the grammar is highly constrained with the intension of supporting specific and targeted limited domain applications.

The performance of this speech recognition system could have even been better but during alignment of the training speech data some of the data were discarded. During alignment, phrases/words that are repeating are discarded, but those words may be necessary for training of the HTK system. As mentioned before, in speech recognition engine development, the training sentences recorded were randomly generated by an HTK tool *HSGen*, so some words or phrases may repeat. The training data and test data were prepared under different conditions. Although most ASR systems are developed with different specifications and objectives this approach performs better than other methods mentioned in literature study chapter 2. The research project results are however not comprehensive and generalizing as we focused mainly on a limited domain speech recognizer for health care application areas.

6.2 Recommendations and Future work

The future work for this type of ASR task includes expanding the domain so that the speech recognizer can be used in more involved health-care applications. This would of course require much more time as this would need collection of more domain specific speech training data and speech test data.

Expanding the allowable syntax structures toward typical sentence structures encountered in a health care consultative episode would also make the recognition system more relevant and applicable to the general health care community.

The process of tying grammatical language model structures with bigram and trigram language models promises to yield better and more robust speech recognition results.

References

[1] Davis, K.H., Biddulph, R. and Balashek, S., 1952. "Automatic recognition of spoken digits"; *Journal of the Acoustical Society of America*, **24**(6):637- 642.

[2] Rabiner, L. and Juang, B., 1993, Fundamentals of Speech Recognition, Prentice Hall, ISBN: 0130151572.

[3] Franco, H., Weintraub, M, and Cohen, M., 1997, "Context Modeling in a Hybrid HMM - Neural Net Speech Recognition System, *Proceedings of the International Conference Of Acoustic, Speech and Signal Processing (ICASSP)*, Atlanta, GA, pp. 2089-2092.

[4] Huang, X. et al. 2001, Spoken Language Processing, Prentice Hall.

[5] Seltzer, M.L., 2007, "Training Wideband Acoustic Models Using Mixed-Bandwidth Training Data for Speech Recognition", *IEEE Transactions on Audio, Speech and Language Processing* Vol. 15, No. 1, pp. 235245.

[6] Mane, A., Boyce, S., and Karis, D., 1996, "Designing the User Interface for Speech Recognition Applications", *SIGCHI bulletin 28*, pp. 29-34.

[7] Rahman, I.A.M.A., 1997, "Arabic Speech Recognition Using Hidden Markov Models", unpublished PhD dissertation, University of Khartoum.

[8] Lopes, L.R., 2009 "An Artificial Intelligence Approach to Improving Speech Recognition", Masters Thesis, University of Cape Town.

[9] Vizitiu, C.I, Munteanu, D.P., 2008, "Robust Romanian language automatic speech recognizer based on multistyle training", *WSEAS Transactions on Computer Research*, Volume 3 Issue 2.

[10] Kim, C. and Stern, R., 2009, "Feature Extraction for Robust speech Recognition Using a Power-Law Nonlinearity and Power-Bias Subtraction, *Interspeech*, Brighton, pp. 2598– 2601.

[11] Ziolko, B., 2009, "Speech Recognition of Highly Inflective Languages", Published Masters Dissertation, University of York. United Kingdom.

[12] Muhirwe, J., 2005 "Automatic Speech Recognition: Human Computer Interface for Kinyarwanda Language", Unpublished Masters Thesis, Makerere University, Faculty of Computing and Information Technology.

[13] Gadde, V.R.R., Stolcke, A., Vergyri, D., Zheng, J., Sonmez, K. and Venkataraman, A., 2002, "*Building an ASR system for noisy environments: SRI'S 2001 SPINE evaluation system"*, Proc of International Conference on Spoken Language Processing, Vol.3, pp. 1577 -1580.

[14] Rabiner L.R., and Wilpon, J.G.,1979, "Considerations in applying clustering techniques to Speaker-independent word recognition", *Journal of Acoustic Society of America 3*, pp. 663-773.

[15] http://www.w3.org/TR/Charmod - last accessed 13 January 2011.

[16] *http*://www.repository.VoxForge1.org last accessed 13 January 2011.

[17] http://www.org.tr/200/note-Voice-XML-20000505 last accessed 13 January 2011.

[18] *http*://www.repository.VoxForge1.org last accessed 13 January 2011.

[19] Arisoy, A., 2008, "Statistical Language Modeling for Automatic Speech Recognition of Agglutinative Languages", Published Masters Thesis, Tallinn University of Technology.

[20] Jelinek, F., 1998, "Statistical Methods for Speech Recognition", Elsevier, ISBN: 0262100665.

[21] Dong, Y., Deng, L., and Acero, A 2009, "A Novel Framework and Training Algorithm for Variable-Parameter Hidden Markov Models", *IEEE Transactions on Audio, Speech and Language Processing*, Vol. 17, No. 7, pp. 1348-1360.

[22] Dong, Y., Deng, L., and Acero, A., 2009, "Using Continuous Features in the Maximum Entropy Model in Pattern Recognition Letters", Elsevier, Vol.30, No. 8, pp. 1295-1300.

[23] *http*://www.repository.VoxForge1.org last accessed 13 January 2011.

[24] http://htk.eng.cam.ac.uk last accessed 25 January 25, 2011.

[25] http://www.adobe.com /products/audition/ last accessed 25 January 2011.

[26] Modiba, T.M., 2005 "Aspects of Automatic Speech Recognition (ASR) With Respect To Northern Sotho" Masters Thesis, University of Limpopo, Turfloop.

[27] Rosenfeld, R., 2000, "*Two decades of statistical language modeling: where do we go from here?"* Proc of IEEE. Vol.88, pp. 1270-1278.

[28] Stefan, B., and Denisa, B., 2003, "*Advances in automatic speech recognition by imitating spreading activation*" in Text, speech and dialogue International conference no 2807, Ceske Budejovice, Czech Republic, Vol. 6, pp. 158 -164.

[29] Wang, S., Greiner, R., Schuurmans. D., 2005, and Cheng, L, "Exploiting Syntactic, Semantic and Lexical Regularities in Language Modeling via Directed Markov Random Fields", *in Proceedings of the 22nd International Conference on Machine Learning*, Bonn, Germany, pp. 948-955.

[30] Bellegarda, J.R., "Large vocabulary speech recognition with multi-span statistical language models" *IEEE Transactions on Speech and Audio Processing*, **8**(1), pp. 76 - 88, 2000.

[31] Young, S., Kersaw, J., Ollason, D., Valtchev, V. and Woodland, P.C, "The HTK Book, Machine Intelligence Laboratory", Department of Engineering, University of Cambridge, UK, 2009.

[32] Manamela, M.J.D. and Botha, E.C., 2001, "C*reating a Northern Sotho telephone speech database for the training of automatic speech recognizers*", 12th Annual Symposium of PRASA, Franschoek, South Africa.

[33] Frost, R., 2004, "*An investigation of Grammar Design in Natural Language Speech Recognition*", Unpublished PhD Thesis, University of Windsor.

[34] Bringert, B., 2007, "Compiling Grammar-based Speech Application Components", Unpublished PhD thesis, Goterborg University, Dept. of Computer Science and Engineering.

[35] Lopes, L., 2008, "*A Software Agent for Detecting and Correcting Speech Recognition Errors Using a Knowledge Base*." In Southern African Telecommunications Networks and Applications Conference, South Africa, Durban.

Appendix A

Test Sentences

The following sentences were used to test the HTK ASR system. The sentences were recorded by 10 Northern Sotho speakers, with 5 of the speakers being female and the other 5 being male.

1. re go thuša bjang
2. mahlo a ka a bohloko
3. letlalo la ka le bohloko
4. nko
5. letswele
6. leoto
7. hlogo ya ka e bohloko
8. matswele a ka a bohloko
9. leihlo la ka le bohloko
10. mahlo
11. maswafo
12. mokokotlo wa ka o bohloko
13. matswele
14. matswele a ka a bohloko
15. nko ya ka e bohloko
16. nko
17. letheka
18. lerapo
19. matswele a ka a bohloko
20. hlogo ya ka e bohloko
21. nko ya ka e bohloko
22. mala a ka a bohloko
23. hlogo
24. matswele a ka a bohloko
25. mahlo

26. mala
27. mahlo
28. leoto
29. letlalo la ka le bohloko
30. leihlo
31. meno a ka a bohloko
32. mahlo
33. meno a ka a bohloko
34. matswele
35. maoto
36. mahlo
37. maswafo
38. matswele
39. leoto la ka le bohloko
40. mala a ka a bohloko
41. kgara
42. maoto a ka a bohloko
43. meno a ka a bohloko
44. mogolo
45. lerapo
46. maswafo a ka a bohloko
47. kgara
48. lerapo
49. maswafo
50. leoto
51. lerapo
52. molomo
53. pelo ya ka e bohloko
54. matswele a ka a bohloko
55. leoto la ka le bohloko
56. letheka la ka le bohloko
57. mogolo
58. molomo wa ka o bohloko
59. leihlo
60. nko
61. maoto a ka a bohloko
62. molomo wa ka o bohloko

63. mogolo
64. pelo
65. matswele
66. letswele
67. leoto
68. mala a ka a bohloko
69. letswele la ka le bohloko
70. kgara ya ka e bohloko
71. letswele la ka le bohloko
72. mokokotlo wa ka o bohloko
73. letswele la ka le bohloko
74. kgara ya ka e bohloko
75. meno a ka a bohloko
76. pelo ya ka e bohloko
77. matswele
78. kgara
79. mala a ka a bohloko
80. re go thuxa bjang
81. maoto a ka a bohloko
82. lerapo
83. molomo wa ka o bohloko
84. matswele a ka a bohloko
85. mala
86. kgara ya ka e bohloko
87. leihlo la ka le bohloko
88. hlogo ya ka e bohloko
89. maoto a ka a bohloko
90. pelo
91. leoto
92. molomo wa ka o bohloko
93. mala
94. matswele
95. letheka la ka le bohloko
96. maswafo a ka a bohloko
97. lerapo
98. letswele la ka le bohloko
99. nko ya ka e bohloko

100. hlogo
101. mahlo
102. molomo
103. re go thuša ka eng
104. maswafo a ka a bohloko
105. nko
106. re go thuša ka eng
107. lerapo la ka le bohloko
108. letheka
109. molomo
110. marapo
111. hlogo
112. mogolo
113. molato ke eng
114. kgara ya ka e bohloko
115. matswele a ka a bohloko
116. mokokotlo wa ka o bohloko
117. lerapo la ka le bohloko
118. leihlo
119. molomo
120. lerapo la ka le bohloko
121. matswele
122. maoto a ka a bohloko
123. maswafo a ka a bohloko
124. nko ya ka e bohloko
125. letlalo la ka le bohloko
126. mokokotlo wa ka o bohloko
127. hlogo
128. letheka
129. kgara ya ka e bohloko
130. re go thuša bjang

Appendix B

Monophones

The following are Northern Sotho monophones as used in this research project.

a
b
bj
d
e
ee
f
fs
g
h
hl
i
j
k
kg
kh
l
m
n
ng
ny
o
p
ph
psh
pš
py
r
s
sil
t
th
tl
tlh
ts
tsh
tš
tšh
u
v
w
x
y
z