CREATION OF A PRONUNCIATION DICTIONARY FOR AUTOMATIC SPEECH RECOGNITION - A MORPHOLOGICAL APPROACH

by

Mpho Caselinah Nkosi

MINI-DISSERTATION

Submitted in partial fulfillment of the requirement for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

in the

FACULTY OF SCIENCE AND AGRICULTURE

(School of Mathematics and Computer Sciences)

at the

UNIVERSITY OF LIMPOPO (Turfloop Campus)

Private Bag X1106

Sovenga

0727

Supervisor: Mr MJD Manamela

Co- supervisor: 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 previously 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.

Nkosi MC (Ms) [redacted] [redacted] [redacted]

Signature: ____________________ **Date:** ____________________

APPROVAL

This report has been submitted for examination with my approval as supervisor.

1. Signed: ....Date: ....
   Supervisor: Mr M.J.D. Manamela

2. Signed: ....Date: ....
   Supervisor: Dr N. Gasela

DEDICATION

To my late mom, and my family
Thank you so much for all your support.

ACKNOWLEDGEMENTS

I would like to express my heartfelt gratitude to the Holy Mighty God, for His mercy, power, and for always being my source of strength and wisdom. I wish to extend my sincere gratitude to my Supervisor, Mr. M.J.D. Manamela and Co-Supervisor, Dr. N. Gasela, for their encouragement and support from the initial stages of the investigation to the concluding level of my studies. Their guidance has enabled me to develop broader and wider understanding of Speech Technology.

I am also thankful to my colleagues, friends and the Computer Science Department staff for their constant support, assistance, and making me feel at home and most of all, for the good times we shared.

I acknowledge the contribution of the Northern Sotho National Lexicography Unit at the University of Limpopo (Turfloop Campus), especially Mr. Mphahlele, for their kind help with all the linguistic problems we had and for the accuracy testing of the pronunciation dictionary. I also acknowledge the University of Limpopo Telkom Centre of Excellence for Speech Technology (TCoE4ST) for their financial support, for without their help, this work would not have been possible.

I acknowledge all my lecturers and all my classmates in the Computer Science Discipline for having made my academic and social life comfortable at the University of Limpopo.

To all of you, may the grace of God be upon you and May He bless you exceedingly abundantly.

List of Abbreviations

ASR	Automatic Speech Recognition
CCV	Consonant-Consonant-Vowel
CSIR	Council for Scientific and Industrial Research
CV	Consonant-Vowel
DICTMAKER	Dictionary Maker
HMMs	hidden Markov models
HTK	Hidden Markov Model Toolkit
MFCC	Mel Frequency Cepstral Coefficient
MLF	Master Label File
OOV	Out-Of-Vocabulary
SLT	Speech and Language Technology
WER	Word Error Rate
SATNAC	Southern Africa Telecommunication Networks and Applications Conference
FSA-PG	Faculty of Science and Agriculture Post-Graduate Research Day
ISIP	Institute for Signal and Information Processing
SPHINX	Speech Recognition toolkit

Table of Contents

LIST OF FIGURES

LIST OF TABLES

SEQUENCE OF LISTS

ABSTRACT

Pronunciation dictionaries or lexicons play an important role in guiding the predictive powers of an Automatic Speech Recognition (ASR) system. As the use of automatic speech recognition systems increases, there is a need for the development of dictionaries that cover a large number of *inflected word forms* to enhance the performance of ASR systems. The main purpose of this study is to investigate the contribution of the morphological approach to creating a more comprehensive and broadly representative Northern Sotho pronunciation dictionary for Automatic Speech Recognition systems.

The Northern Sotho verbs together with morphological rules are used to generate more valid inflected word forms in the Northern Sotho language for the creation of a pronunciation dictionary. The pronunciation dictionary is developed using the Dictionary Maker tool. The Hidden Markov Model Toolkit is used to develop a simple ASR system in order to evaluate the performance of the ASR system when using the created pronunciation dictionary.

1 Introduction

In this chapter:

- *The background review in speech and language technology*
- *Literature review on Automatic speech recognition technology*
- *Current state of the Automatic Speech Technology*
- *The focus of the study*
- *The objectives of the study*
- *Related work*
- *The organization of the study*

1.1 Background

Speech and language technology has a special place in the use of computational devices for information and communication processing worldwide. It is an information technology designed to specifically deal with human language. Speech and language technology includes, amongst other sub-disciplines, recognition (Automatic Speech Recognition), synthesis (Text-To-Speech) and translation. This technology has given people an opportunity to use and manipulate information in a more natural manner and in general, it also contributes to the advancement of broad intercultural and linguistic understanding. It enables users to communicate with a computer in their first language (that is home language). It makes it possible for people to use a simple and widely used device such as a mobile device to access information, use email system or even do online transactions in their home language. The world Wide Web and mobile devices are now using many applications in speech and language technology. For example, spoken interfaces for devices such as *iphone*, creating text from speech for *dictation* and *machine translation* from one language to another. This technology might be one of the solutions in the near future that can also help developing communities to reduce the digital-divide that exists between urban and rural areas (Furuholt and Kristiansen, 2007).

Speech and language technology relies heavily on the comprehensiveness of speech and language resources such as pronunciation dictionaries, annotated audio speech corpora, text corpora, intonation enabled systems for tonal languages, and interpretation pronunciation. However, these resources pose a significant challenge in developing this technology, especially for the under-resourced languages of the developing countries. For such languages, there may be limited or no available usable resources, and creating them from scratch is usually very difficult and costly.

As access to information technology is one of the most important requirements for social development these days, the use of language and speech technology has further become a necessary technological development for linguistic resources. In order to ensure that people have sufficient access to information or linguistic resources expressed in their home language, these resources should be translated into digital electronic representation. For example, the Christian Bible is now available in digital electronic audio format in many languages. Apart from the electronic audio format, the linguistic resources may include dictionaries, grammatical rules, terminological thesaurus, etc.

1.2 Literature Review on Automatic speech recognition

As one of the speech and language technology, Automatic Speech Recognition (ASR) is a process through which a computer converts an acoustic signal captured by a microphone to a set of words (Mengjie, 2001). The ASR technology has been around for quite some time now. This technology began in the 1930s when AT&T Bell laboratories developed a speech recognition device with an objective of creating a machine that can imitate human behaviour, with respect to speaking naturally and responding properly to spoken language (Rabiner et al.,2005). Ever since then, speech technology researchers have put much focus on a series of investigations that would help and improve the development of a much better speech recognition system. As of today, the ASR technology has improved tremendously and its capability for recognizing human speech has progressed significantly. Its performance has been developed by the use of many applications ranging from a simple digit recognition system to a large vocabulary broadcast news transcription (Osterndorf et al., 2002). The earliest ASR technology used the template-based approaches and the knowledge-based approaches as speech recognition techniques. On the latter recognition technique, an expert knowledge about variations in speech is hand coded into the system. However, the expert knowledge is difficult to obtain and to use successfully (Zue et al., 1986). When using the template based approaches to speech recognition, unknown speech is compared against a set of pre-recorded words in order to find the best match. The only problem with this technique is that the pre-recorded speech is fixed and as such, speech can be represented by using many templates per word (Wilpon et al., 1979). Today's success of the ASR technology is supported by the existence of the powerful statistical modeling techniques in which speech variations are represented statistically using the statistical learning procedures such as the hidden Markov models (HMMs).

1.2.1 The current state of the ASR technology

The ASR technology is currently employed for daily usage in several places. The automated phone answering services are the most commonly used applications today and they attain almost 100% recognition accuracy even with untrained users. These automated services are also employed in retail banking in order to provide an outstanding customer experience and to reduce fraud and loss (Development Management Group, 2010). In Mobile devices, the Android's smart phones use the voice search services that are based on the ASR technology. It has been announced by the Nuance Company that the use of the ASR technology in the legal and medical transcription has reduced the costs by 30% (Nuance, 2010). The Google Company has now employed the ASR technology to offer voice search services and for use in the Google voice translation from one language to another. Chelba et al., (2010) state that although the ASR technology has advanced, it is still not a solved problem due to noise robustness, speaker variability and language model mismatches. ASR still has high error rates under noisy conditions. An ASR system with a smaller speech database during its training phase does not usually cope with the incident of speech variability from one person to another.

*1.*3 **Focus** ***of the research project***

An all-the-time goal of speech recognition research has been to come up with technologies that will enable voice communication between humans and computers. As a result, a conversion that will make it possible for the transcription of human speech into a text that a computer can represent in user readable format is required. ASR is one such conversion interface system that can make this goal achievable. The success of an ASR system relies heavily on the availability of a well-designed pronunciation dictionary. A pronunciation dictionary forms part of the key components of speech recognition systems. It plays an important role in *guiding the predictive powers* of an ASR system. A pronunciation dictionary that covers a large number of inflected words can enhance the performance of the ASR system.

The research work in this report focuses on the creation of a pronunciation dictionary (lexicon) for automatic speech recognition systems using a morphological approach on the verb base-forms in Northern Sotho - one of the eleven South African official languages spoken by nearly 10% of the RSA population as a home language.

1.4 Objectives of the study

In this study, the goal is to determine whether the morphological approach is useful in creating a more comprehensive and broadly representative Northern Sotho pronunciation dictionary that can be incorporated with ASR systems in Northern Sotho. In addition, we develop an ASR engine for Northern Sotho that can be used in future projects over and above being used to evaluate the performance of the ASR system when using the pronunciation dictionary.

1.5 Related Work to morphological Approach

In the last few years, related research work has been conducted in the morphological approach to aim improve *word coverage*. Sunitha et al. (2009) showed that morphological analysis is a significant step of natural language processing for the highly inflectional languages. In their approach, words were generated by combining the stems belonging to a particular stem bag with the corresponding suffix bag. These words were treated as the training data for the next set of new samples for the

Hindi language. The results of the experiment indicate that using a morphological analyzer with a limited word list can result in generating a large number of words which would improve the range of coverage.

Biadsy et al. (2009) used morphological analysis and disambiguation tool together with linguistically-based pronunciation rules to improve an Arabic pronunciation dictionary. In this approach, pronunciation rules were modified and a new set of pronunciation rules were introduced generating additional word variants. The pronunciation dictionary that covers all words in the orthographic transcription of the training data was developed. For each word in every utterance, morphological analysis and disambiguation tool (tool) was used to disambiguate the word, based on its context in transcription. The tool outputs all possible fully diacritized morphological analyses of each word ranked by their likelihood. The highest ranked diacritization of each word was mapped to a set of pronunciations.

1.6 Organization of the thesis

This mini-dissertation is divided into seven chapters. The next chapter will discuss the structure of verbs in Northern Sotho. Chapter three will examine the morphological approach that is used in coming up with the word forms used in the development of the pronunciation dictionary. The alignment and training of the dictionary is discussed in chapter four. Chapter five describes the hidden Markov model and the acoustic model used for the development of a simple ASR system in this work. The results of applying the morphological approach during the training of the dictionary and incorporating it in the ASR system are discussed in chapter six. Finally, conclusions derived from these experiments, discussions and future work will be discussed in chapter seven.

2 Verbs in Northern Sotho

In this chapter:

2.1 Introduction

Northern Sotho is one of South Africa's official languages that is spoken predominantly in Limpopo Province. Generally, it is spoken in the North-Eastern parts of Pretoria, in parts of Gauteng Province, Limpopo and Mpumalanga (Joffe, 2011). According to the official Census 2001, Northern Sotho is spoken by at least 9% of the total RSA population. This language is closely related to other languages in the Sotho language group, that is, Southern Sotho and Setswana. A speaker of one of the three languages is usually able to understand most of what another speaker of one of the other two language is saying.

This chapter looks briefly at the structure of the verbs in the Northern Sotho language and the types of verbs used.

2.2 What is a verb in Northern Sotho?

In Northern Sotho, a verb is a word or part of the word that communicates an action or state. According to Poulos & Louwrens, (1994), a verb consists of a number of elements that make up a word that represents the constituent parts of a word. They further point out that these elements maybe, for example, a *subject concord* which refers to the subject of the verb, a *tense marker* which shows a particular tense, an *object concord* which refers to some or other objects, a *verb root* which communicates the basic meaning or state, a *vowel ending* which comes at the end and which sometimes gives an indication of the tense of the verb. They refer to these elements as morphemes. However, it is noted that not all of these elements always occur in a verb. The verb root on the other hand, is a compulsory part of each and every verb. However, a verb root does not form a complete verb on its own. It is combined with prefixes and suffixes to make it complete (this is explained in the next section). A verb root can take on different forms.

Root Structure	**Root is made of**	**Example**
C	consonant only	-y- (go *to)*, -psh- (*dry up).*
CVC	consonant + vowel + consonant	-rom- (*send)*, -lem- (*plough)*
VC	Vowel +consonant	-ag- (*build),* -om- (*become dry)*
CVCVC	consonant + vowel + consonant + vowel + consonant	-sepel- (*walk),* -rumol- (*rovoke)*
CVVC	consonant + vowel + vowel + consonant	-laol- (*command),* -lael- (i*nstructs)*
VCVC	vowel + consonant + vowel + consonant	-arab- (*answer),* -opel- (*sing)*
VCV	vowel + consonant+ vowel	-ape- (*cook),* -iti- (*beat).*
CVCV	consonant + vowel + consonant + vowel	-tlale- (*report),* -gege- (*mock at)*
CVCVCVC	consonant + vowel +consonant + vowel + consonant +vowel + consonant	-hlokomel- (*attend to),* -nanabel- (*stalk).*
VCVCVC	vowel + consonant + vowel + consonant + vowel + consonant	-edimol- (*yawn),* -ethimol- (s*neeze)*
CVCVVC	consonant + vowel + consonant + vowel + vowel consonant	-kadiel- (*suspend)*
CVVCVC	consonant + vowel + vowel + consonant +vowel + consonant	-goelel- (*yell)*

Table 2.1: The structures of the verb roots in Northern Sotho (adopted from Poulos and Louwrens, (1994).

Table 2.1 describes the different forms which a verb root may take. In the given examples, the verb roots are written with hyphens on both sides to show that prefixes and suffixes may be added. The letter C represents a consonant and V represents a vowel.

Some of the examples given in Table 2.1 have two or three letters in orthography (e.g., -psh- given on the C roots structure); these in fact represent **complex sounds** (Price and Gee, 1988). Appendix A shows all the consonants in the Northern Sotho language.

2.3 Types of verb roots

- **Simplex roots**
  These types of roots do not include any extensions and further, they are divided into three types. The first type is the *primitive roots*. Primitive roots are not drawn from any other form or part of speech (Poulos and Louwrens, 1994). Second type is the *derivative roots*. These are the opposite of the primitive roots. They are drawn from other parts of speech or word categories by adding some affixes. The last type is the adopted roots. *Adopted roots* are adopted from other languages.
- **Extended roots**
  Extended roots include extensions that modify the basic meaning of a root and form a new root. (e.g., the extension -an- may be added to the root –tseb- “*know*” to form an extended root -tseban- “*knowing each other*”). There are many extensions that can be added to a verb root and each is connected with its own meaning or meanings.
- **Reduplicated roots**
  This is the reduplications of either simplex or the extended roots. In most cases, these roots represent an action that is performed repeatedly or frequently. e.g., “emaema” (*going up and down*).

The above-mentioned types of roots represent the *morphological types of verb roots*. There are *semantic types* of verb roots that deal with the basic meaning of the verb roots. The semantic types will not be looked at here as they do not form part of this study.

3 Morphological Approach

In this chapter:

- *Introduction*
- *Morphological structure of the Northern Sotho verbs*
- *The affixes in the Northern Sotho language*
- *The formation of verbs in Northern Sotho*
- *Morphological rules*

3.1 Introduction

In general, morphology is a very broad subject on its own. In the context to this study, morphology refers to the identification, analysis, and description of the structure of morphemes of the language. In this chapter, the morphological structure of the Northern Sotho verbs is identified and analyzed. The base verbs will be generated using syllable structures. The morphological rules will be developed to analyze the generated base verbs and to generate new valid word forms. Figure 3.1 below summarizes the morphological approach that is used in this study.

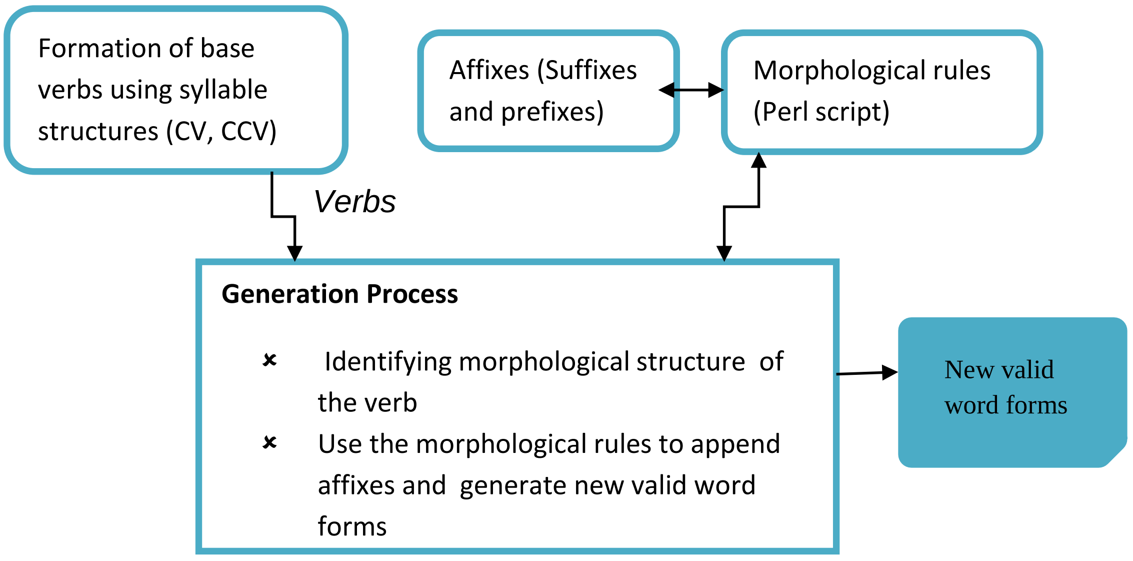

Formation of base verbs using syllable structures (CV, CCV)
Affixes (Suffixes and prefixes)
Morphological rules (Perl script)
Verbs
Generation Process
Identifying morphological structure of the verb
Use the morphological rules to append affixes and generate new valid word forms
New valid word forms

Figure 3.1: The Morphological approach

3.2 Formation of verbs

Northern Sotho is regarded as an agglutinative tonal language in that, nouns and verbs are formed by means of roots and affixes (prefixes and suffixes). Poulos & Louwrens, (1994), state that “most of the roots are bound to morphemes because they cannot form words by themselves but need one or more affixes to complete the word”.

In the morphological approach, the consonants and vowels in Table 3.1 and Table 3.2 respectively, are used to formulate base verbs and to yield a good and diverse root structure. Each of the consonant is combined with every vowel to form syllable structures. For each combination, base verbs beginning with the syllable structure, consonant vowel (CV), are formed. For example, for the consonant ‘*b’*, these combinations of CV syllable structure are represented as: *ba, be, bi, bo, bu.* The base verbs starting with these combinations are manually formed. For example, with the syllable structure *‘ba’* the following are some of the base verbs that were formed:

- *baka – fight for something*
- *bala – read*
- *bapola – crucify*

b	j	ph	tš	š
bj	k	psh	tsh	
d	kg	pš	tšh	
f	kh	pšh	tl	
fs	l	r	tlh	
fš	m	s	w	
g	n	t	y	
h	p	th	ng	
hl	ph	ts	ny	

Table 3.1: consonants used to form Northern Sotho base verbs

a	*e*	*i*	*o*	*u*

Table 3.2: Vowels in Northern Sotho

Although some of the consonants in Table 3.1 consist of two or three letters, they do not form the CCV or CCCV syllable structures but the CV syllable structure because they represent *complex consonants* in Northern Sotho (Price & Gee, 1988).

3.3 Morphological structure of the verb in Northern Sotho

As already discussed in Chapter 2, a verb is a word that shows an action that is performed by the doer. There is a connection between the doer and the verb that is brought by a word that connects the two together. For example, "*bana ba a bala"* (children are reading*).* From this sentence, '*ba*' is the connector that connects doer *(bana)* and the verb *(bala).* In Northern Sotho, the most basic form of a verb can consist of the head (prefix), body (root) and the ending (suffix). The root always forms the lexical core of the verb. Kruger (2006) defines a verb root as "part of a word that does not include a grammatical morpheme; cannot occur independently as in the case of words; constitutes the lexical meaning of a word and belongs to an open class".

3.3.1 Common *Prefixes of the Northern Sotho Verb*

The prefix that is mostly used in Northern Sotho is an *–i.* This prefix is placed just before the verb root. In most cases, the verb that begins with this prefix shows an action which the doer does on himself or herself. For example, ithata (loving oneself) which is derived from the verb *rata* (love) as *i-rata (loving oneself)*. However, when this prefix is placed before the verb root, it causes the sound of the first letter of the verb root to change *(e.g., rata* (love) – *ithata* (loving oneself), *r* changed to *th* when the prefix –*i* was placed before the root –*rat-*).

Other prefixes that can be placed before a verb root are *-n* and *-m.* When these prefixes are placed before a verb root, they show an action that is performed by someone on the doer. For example, *rata* (love) - *nthata* (loves me); bona (see) – *mpona* (sees me). Like the prefix *–i,* these prefixes also cause the first sound of the verb root to change*(r → th, b →p).*

3.3.2 Suffixes of the Northern Sotho Verbs

Suffixes are used as verb endings. Extended roots that were mentioned in the previous chapter are usually formed by adding suffixes to a verb root in order to show different actions/meanings. There are several suffixes that are used in Northern Sotho. This study looks at ten different suffixes that can be added to a verb root to show different actions.

- **Applied suffix -el**

  Verb roots that include this suffix show an action that the doer does for someone or something. (e.g., from the verb *rata (to love), the suffix –el is* added to the root *–rat-* to form *"rat**el**a"* – love for something/someone).

- **The neuter suffix –eg**
  The suffix –eg is used in a verb to show an action that can be possible or can be done or is done (e.g., from the verb *rata (to love), the suffix –eg is* added to the root *–rat-* to form *rat**eg**a* – lovable/likable).
- **The causative and assistative suffixes: –iš and –išiš**
  When the –iš suffix is added to a root, it may show that the doer causes someone/something to perform the action stated in the verb or it may shows that the doer assists someone/something to perform an action stated in the verb (e.g., from the verb *rata (to love), the suffix* –iš *is* added to the root *–rat-* to form *rat**iš**a* – cause to love someone/thing). On the other hand, the suffix –išiš is added to a root to show that whatever that is done, is done with enthusiasm.
- **The reversive suffix –oll**
  The roots which are obtained with this suffix may take objects (e.g., from the verb *dira (do), the suffix –oll is* added to the root *–dir* to form dir**oll**a –redo).

- **The reciprocal suffix –an**

  The verb roots that include this suffix show an action that is performed by people involved (e.g., from the verb *rata (to love), the suffix –an- is* added to the root *–rat-* to form rat**an**a – love one another).

- **The iterative suffix –ak**

  The verb roots with this suffix show that an action is done again and again (e.g., from the verb *gata (kick), the suffix –ak is* added to the root *–gat-* to form gat**ak**a – kick repeatedly).

- **The passive suffix –w**

  The passive suffix indicates the passive form of the verb (e.g., from the verb *rata (to love), the suffix –w is* added to the root *–rat-* to form rat**w**a – be loved).

- **The perfect suffix –il-**

  The perfect suffix, when included in a verb, shows an action that has been performed (e.g., from the verb *rata (to love), the suffix –il is* added to the root *–rat-* to form *rat**ile*** – loved/liked).

3.4 Morphological Rules

The most important part that expresses the meaning is the verb root. In Northern Sotho language, *the verb root and an inflectional beginning or ending* are the ones that convey the unique meaning of a word. The basic meaning of a verb can be modified in many different ways by adding suffixes and prefixes. The base verbs that were formed in section 3.2 are analyzed and morphological rules depending on these base verbs are developed. Each rule checks the root of the verb and modifies the meaning of the verb by adding a suffix/prefix on the context of the verb root to yield a new and valid word form.

A Perl script is used to develop the morphological rules. When executed, the script is initialized with a verb. It then analyzes the verb and modifies it by appending the necessary prefixes/suffixes to form new valid word forms in Northern Sotho. For this study, the script uses the **ten suffixes and the prefix -*i*** discussed in the previous section. Listing 3.1 depicts the script used to generate new valid word forms when given a verb. It analyzes a verb at a time and checks as to whether the verb meets the conditions required to generate new word forms. For example, to form a new word using a suffix –w-, if the verb ends with -pa, then –pa is replaced by –pša. Table 3.3 shows some of the conditions and rules that were used to analyze the verbs and generate new word forms. Some verbs meet all the required agglutination conditions whereas others can meet only a few or nothing at all. In case of nothing at all, the script outputs blank results and in a few, it outputs few word forms that were legally generated. In most cases, when a verb does not meet all the required conditions, it is due to some restrictions on the combinations of some of the inflected word meanings. Listing 3.2 shows the results obtained when the script was initialized with verb *rata –to love.* The script generated ten new valid word forms in Northern Sotho although it was supposed to generate eleven word forms. The reason for this is attributed to restrictions on the inflected word meaning. The script was able to construct new and valid words that are not included in the existing Northern Sotho dictionary. The reason for this exclusion may be attributed to their rare usage in the day-to-day spoken language or the meaninglessness of the constructs.

Table 3.3 lists some of the rules that are applied to a verb to form new word forms. The table shows how sound changes when prefixes/suffixes are added to a verb. Each verb ending in the Prefix/Suffix column is replaced by the corresponding rule in the rule column to form a new word by using a particular prefix/suffix.

Table 3.3: Rules used and applied when analyzing a verb

Prefix/ Suffix	Rule
Suffix –w- : when a verb ends with:	
• *-pa*	*-pa → -pša*
• *-ba*	*-ba → -bja*
• *-fa*	*-fa → -fša*
• *-pha*	*-pha → -pšha*
• *-ma*	*-ma → -ngwa*

• *-ta*	*-ta → -twa*
• *-ka*	*-ka → -kwa*
• *-la*	*-la → -lwa*
Suffix –an- : when a verb ends with an *-a*	*-a → - ana*
Suffix –eg- : when a verb ends with an *-a*	*-a → -ega*
Suffix –el -: when a verb ends with:	
• *-ja,-ga,-ma, -ta,-na*	*-a → -ela*
• *-nya, -ša*	*-a → -etša*
• *-tša*	*-tša → -letša*
• *-la*	*-la → -lela*
Suffix –iš- : when a verb ends with:	
• *-ra, -ma*	*-a → -iša*
• *-ya*	*-ya → -iša*
• *-la*	*-la → -iša*
• *-ga*	*-ga → -ša*
• *-na*	*-na → -ntšha*
• *-nya*	*-nya → -ntšha*
• *-tla*	*-tla → -tliša*
Suffix –ak- : when a verb ends with -*a*	*-a → -aka*
Prefix –i- : when a verb begins with:	
• *-r*	*-r → -th*
• *-b*	*-b → -p*
• *-l*	*-l → -t*
• *-d*	*-d → -t*
• *-f*	*-f → -ph*
• *-o*	*-o → -ko*
• *-a*	*-a → -ka*

```
#!/usr/bin/perl –w

print"Please enter a verb to process:";            # REQUESTING A VERB INPUT FROM THE USER

chop($string=<STDIN>);                         #CAPTURING THE ENTERED INPUT AND STORES AS STRING

$originalverb=$string;                              # ASSIGNINIG THE STRING TO A NEW VARIABLE

$ledirwi=$string;

$fragment=substr $ledirwi,-2;                                  #ASSIGNING THE VERB ENDING TO A VARIABLE
FRAGMENT

print "**************************************************\n";

print"Words generated from this verb "; print "<$originalverb>\n";    # PRINTING THE ALL THE NEW
#GENERATED WORDS FROM THE ENTERD VERB

#_________________________________________________________________

#Ledirwa  # ANALYZING THE VERB TO GENENRATE A NEW WORD BY USING THE  PASSIVE SUFFIX -W-

if ($fragment eq 'ta')                                        #CHECKING IF THE VERB ENDING IS  'ta'

{ substr($ledirwi,index($ledirwi,'ta'),2)= 'twa';  # IF THE VERB ENDING IS 'ta', THEN 'ta' REPLACED BY #'twa'
TO GENERATE A NEW WORD

 print"<$ledirwi>\n";

}elsif ($fragment eq 'ba')

{  substr($ledirwi,index($ledirwi,'ba'),2)= 'bja';

  print"<$ledirwi>\n";   }

elsif ($fragment eq 'sa')

{  substr($ledirwi,index($ledirwi,'sa'),2)= 'swa';

  print"<$ledirwi>\n";}

elsif($fragment eq 'pa')

{ substr($ledirwi,index($ledirwi,'pa'),2)= 'pxa';

  print"<$ledirwi>\n";}
```

Listing 3.1: Script used to analyze and form new valid words

```
Words generated from this verb <rata>

<ratwa>

<ratana>

<ratega>

<ratela>

<ratixa>

<ratolla>

<ratixixa>

<rataka>

<ratile>
```

Listing 3.2: Results obtained by running the script initialized with verb 'rata'

4 Pronunciation Dictionary Creation

In this chapter:

- *Introduction*
- *Dictionary initialization*
- *Dictionary training*
- *Verification process*
- *Rule extraction process using the Default & Refine Algorithm*

4.1 Introduction

Creating a pronunciation dictionary from scratch is not a difficult task to do. It can be created manually by a linguist or language expert in the target language. But because of the large size of the vocabulary, the manual approach can be time consuming and expensive and therefore it is usually an undesirable option. For this reason, the process has to be partially automated to make it simpler and cost effective.

In this chapter, we look at the algorithm and the procedure that was used to create a pronunciation dictionary (aka lexicon) as one of the speech and language resources. A pronunciation dictionary can be thought of as a mapping of words to their corresponding pronunciations. Such a dictionary is different from an electronic dictionary in that, it does not provide the meanings of the words but only *how they are pronounced* using phoneme representation. In automatic speech recognition, a pronunciation dictionary lists words that are known to the recognizer and helps in building acoustic models for each entry. In the following sections, we will refer to a pronunciation dictionary as a dictionary or a lexicon. Figure 4.1 summarizes the dictionary creation process.

4.2. Initialization

In this study, the dictionary maker (aka dictmaker) tool is used for the training and creation of Northern Sotho pronunciation dictionary. Dictmaker has been developed by the Council for Scientific and Industrial Research (CSIR) Meraka Institute. It is used to monitor the creation of a pronunciation dictionary for a selected language. For a new dictionary creation, dictmaker is initialized with a phoneme set, wordlist and grapheme set.

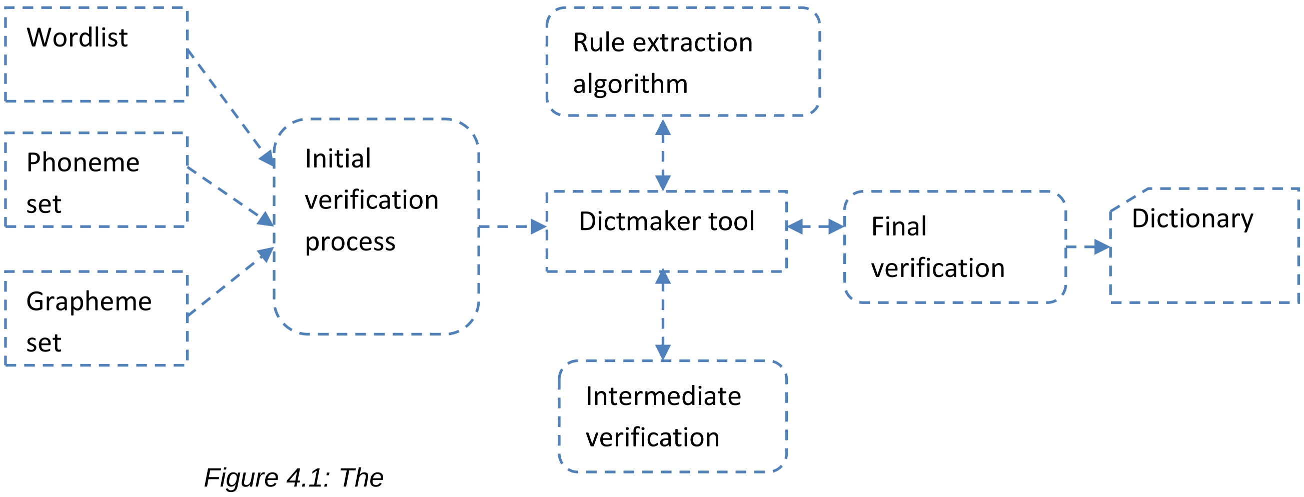

Wordlist
Phoneme set
Grapheme set
Initial verification process
Rule extraction algorithm
Dictmaker tool
Intermediate verification
Final verification
Dictionary

Figure 4.1: The pronunciation dictionary creation process

4.2.1 Wordlist formation

A wordlist is an alphabetically ordered list of words that the dictmaker uses for the training and creation of a dictionary. These are the words that are going to be in the final dictionary with their corresponding pronunciations. The verbs together with the newly generated word forms created using the morphological rules in chapter 3 are used as the wordlist for this study. The list is alphabetically ordered and each word is on a new line.

4.2.2 Grapheme set

A grapheme set is a set consisting of all the letter symbols that are allowed in the Northern Sotho language. As with the wordlist, each grapheme is also on a new line.

a	f	j	n	r	y
b	g	k	o	t	š
d	h	l	p	u	
e	i	m	r	w	

Table 4.1: Grapheme set used for the dictionary training

4.2.3 Phoneme set

A phoneme set consists of a list of unique sounds used in the target language. These sounds are used by the dictmaker to play the audio version of a word during the dictionary training. Consonants and vowels in Table 3.1 and Table 3.2 respectively, are used as the phoneme set. Each phoneme is recorded clearly articulated with some of the inter phone slightly exaggerated.

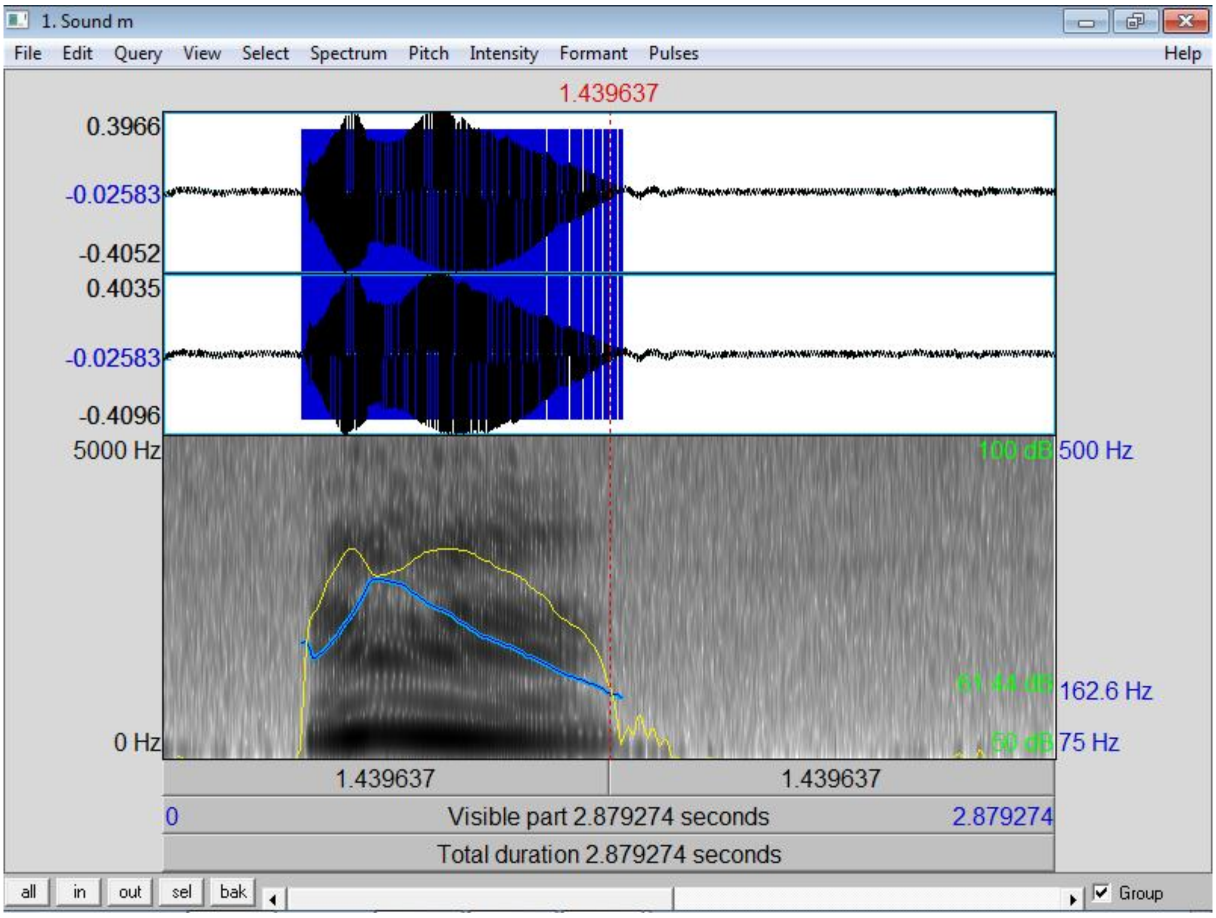


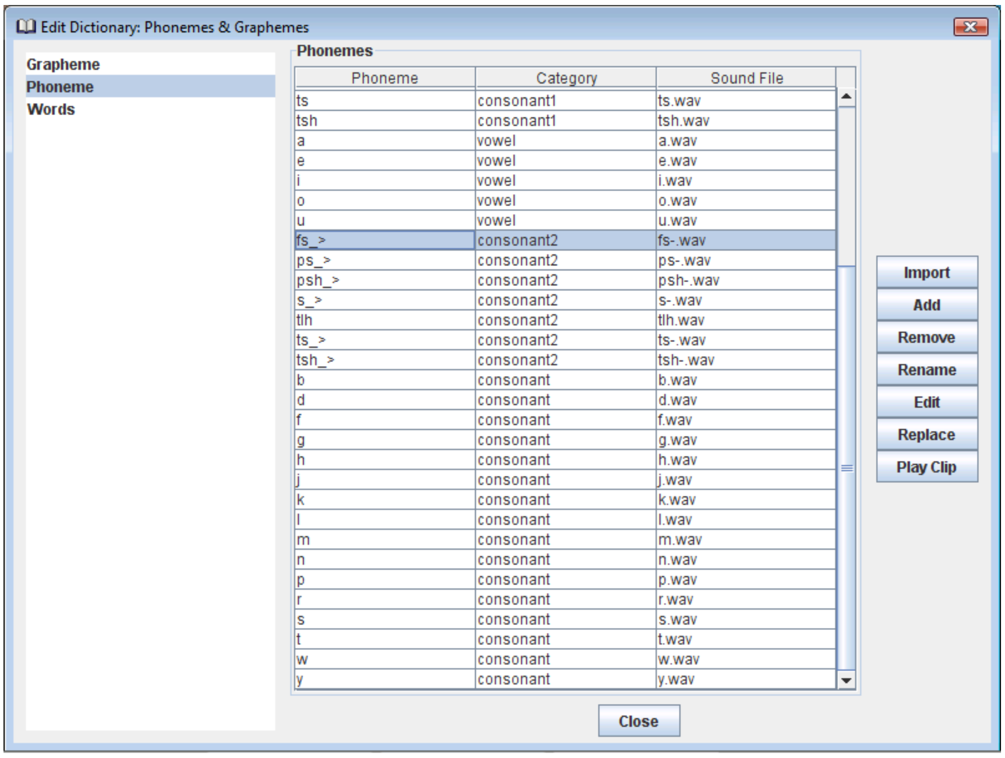
Figure 4.2: speech analysis of the object sound m

The noise disturbances (the “ums” and “ohs”) are manually removed from the start and the end to ensure that the set is normalized and that consistency is maintained. The phoneme set has been recorded using *Praat*[1]. *Praat* is a freeware program designed by the University of Toronto’s Graduate Department of Speech-Language Pathology for the analysis and reconstruction of acoustic speech signals

[1] *Praat* is a system for doing phonetics downloadable at http://www.praat.org

(Van Lieshout, 2004). One user was used to record the set of phonemes. Each phoneme was recorded as a stereo sound at 44 100 Hz sampling frequency.

To normalize and maintain consistency, each sound is analyzed. Figure 4.2 shows the analysis of the sound m. The top panel shows the waveform and the bottom panel shows a spectrogram of the sound signal. Below the bottom panel is the total duration of the sound signal. The pulses that extract the details on voice parameters are indicated by the blue shaded area of the top panel. Such parameters may indicate the jitteriness and the shimmering on the voice of the speaker. The intensity used to control the intensity signal is shown by the yellow line in the bottom panel. The blue line indicates the pitch used on the voice signal. Each validated phoneme sound is saved as a wave file.



Phoneme	Category	Sound File
ts	consonant1	ts.wav
tsh	consonant1	tsh.wav
a	vowel	a.wav
e	vowel	e.wav
i	vowel	i.wav
o	vowel	o.wav
u	vowel	u.wav
fs_>	consonant2	fs-.wav
ps_>	consonant2	ps-.wav
psh_>	consonant2	psh-.wav
s_>	consonant2	s-.wav
tlh	consonant2	tlh.wav
ts_>	consonant2	ts-.wav
tsh_>	consonant2	tsh-.wav
b	consonant	b.wav
d	consonant	d.wav
f	consonant	f.wav
g	consonant	g.wav
h	consonant	h.wav
j	consonant	j.wav
k	consonant	k.wav
l	consonant	l.wav
m	consonant	m.wav
n	consonant	n.wav
p	consonant	p.wav
r	consonant	r.wav
s	consonant	s.wav
t	consonant	t.wav
w	consonant	w.wav
y	consonant	y.wav

Figure 4.3: The phoneme set initialized to dictmaker

The validated phoneme set is initialized to dictmaker. Dictmaker provides categories to sort the phoneme set for ease of use during training. For this study, we used three categories: the consonant - single letter consonants, the consonant1 - two\three letter consonants, the consonant2 - complex consonants, and the vowel. The complex sound consonants are named using an underscore symbol and a greater than sign. For example, in Figure 4.3, the consonant _'*š'*_ as one of the complex consonant is represented as s_>.

4.3 Verification

As discussed by Davel and Martirosian (2009), the verification process is used to facilitate the dictionary creation process and to ensure that some errors are recognized and dealt with earlier before they are compounded.

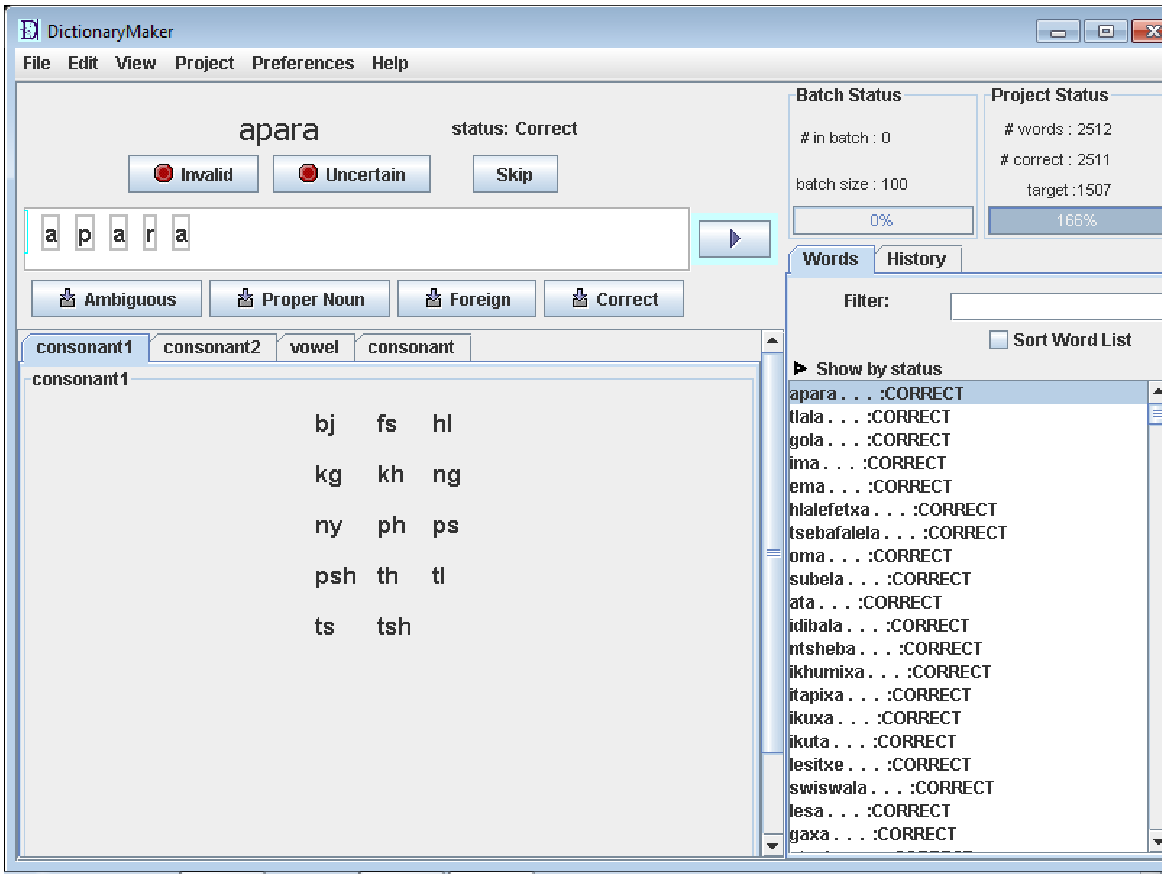
The initial verification process is performed immediately after initialization. This is a very crucial stage, as the errors that are not recognized at this stage might lead to complications in the later stages of the dictionary creation process and might lead to the final dictionary having incorrect word pronunciations. The wordlist is analyzed thoroughly, wherein each word is checked as to whether it is correctly spelled and that it is not repeated unless in the case where the word is referring to different items. The grapheme set is analyzed to check that all the letters are included in the set and that each letter is not mistakenly repeated. The phoneme set is analyzed to check that each phoneme is correctly recorded and that each sound is correctly captured. For example, the phoneme 'bj' was captured as '*vhyee*' instead of *'vhjj'.* The recordings were analyzed with the help from the language expert.

The second stage of the verification process is the intermediate verification. It is performed during the training phase. The system is checked as to whether it can identify errors, and that all the graphemes are included in pronunciations. It also analyzes the usage of the phoneme set.

The last stage of verification is the final verification. Final verification occurs when the dictionary is ready for acceptance. The dictionary is analyzed word by word, to ensure that each word is correctly pronounced. Words that need to be modified are identified and this process is repeated until there are no more words that need any modification. Errors that were identified at this stage were mostly of the words that were incorrectly pronounced, whereas others were mistakenly declared invalid or ambiguous by the developer during training. Such errors might have been made by the developer due to the misunderstanding of the word.

4.4 Training phase

When the dictmaker has been initialized and the initial verification process is successful, the training phase begins. The dictmaker now has all the required sets to start training the dictionary. The developer can either use the '*system select mode'* or '*the user select mode*' to choose the next word for pronunciation. Depending on the developer's choice, the dictmaker predicts the pronunciation of a word and provides the developer with an audio version of the word by using the recorded phoneme set. The dictionary developer acts as the 'judge'.



```
apara                status: Correct

a p a r a

Batch Status          Project Status
# in batch : 0        # words : 2512
batch size : 100      # correct : 2511
                      target :1507
0%                    166%

consonant1
bj   fs   hl
kg   kh   ng
ny   ph   ps
psh  th   tl
ts   tsh

apara . . . :CORRECT
tlala . . . :CORRECT
gola . . . :CORRECT
ima . . . :CORRECT
ema . . . :CORRECT
hlalefetxa . . . :CORRECT
tsebafalela . . . :CORRECT
oma . . . :CORRECT
subela . . . :CORRECT
ata . . . :CORRECT
idibala . . . :CORRECT
ntsheba . . . :CORRECT
ikhumixa . . . :CORRECT
itapixa . . . :CORRECT
ikuxa . . . :CORRECT
ikuta . . . :CORRECT
lesitxe . . . :CORRECT
swiswala . . . :CORRECT
lesa . . . :CORRECT
gaxa . . . :CORRECT
```

Figure 4.4 : Sample dictmaker system training the dictionary

The judge can rule 1) that the pronunciation is correct (meaning that the pronunciation is hundred percent correct), 2) that the pronunciation is ambiguous (meaning that the word has more than one pronunciation), 3) that the word is invalid, 4) uncertain about how the word is pronounced, 5) that the pronunciation is incorrect, 6) that the word be skipped before coming back to it later. In case of a word pronunciation declared being incorrect, the judge provides the correct pronunciation by dragging from the phoneme panel the correct phonemes and replacing the incorrectly provided ones. A new audio version of the pronunciation would then be provided and the judge would act on it. This process is repeated until sufficient dictionary size is reached or until all the words in the wordlist are verified.

When a word pronunciation is declared correct, the system uses the rule extraction algorithm, Default*&Refine algorithm* discussed later, *to* extract a new rule set which will be used to predict the next word pronunciations. The more the rule sets in the system, the more accurate predictions are provided. Dictmaker uses three updating modes to extract new rule sets, the continuous mode, batch mode and the incremental batch mode. In the continuous mode, the system extracts a rule set every time a word is declared correct. The batch mode updates the rule set once the specified batch size is completed. On the other hand, the incremental batch mode is not so different from the batch mode except that it uses an incremental rule update which uses the less accurate version of the rule extraction and performs a full update once a batch size is declared correct. However, the rule set can be updated before the batch size is reached by synchronizing the rule set. During the training phase, the developer can sort the wordlist and show status by showing only the correct, unverified, ambiguous, uncertain or invalid words.

4.5 Default & Refine Algorithm

At the initialization stage, wordlist is initialized to the dictmaker without any pronunciation information. When the training phase begins, the dictmaker has to predict pronunciations for each word in the wordlist so that the dictionary developer can provide a verdict for each prediction provided. This is very crucial because the ability of the dictmaker to predict the pronunciations of the words plays an important part in powering the speed at which the dictmaker learns. To be able to provide predictions, the dictmaker uses the default and refinement algorithm (aka default&refine).

Davel et al. (2008) define default&refine algorithm as “the rule based learning algorithm that was developed as an accurate and efficient pronunciation prediction mechanism for speech processing systems.” The default&refine algorithm is similar to the fuzzy rules in that small training sets can be used to perform rapid generalization. The default&refine algorithm learns efficiently and can provide higher accuracy even when trained on small sets. On the other hand, the number of rules increases exponentially with the number of input variables when using fuzzy rules.

As already mentioned, the Dictmaker is initialized without any pronunciation information and at this stage, the rule set which the Dictmaker uses to predict words pronunciations is empty. The developer has to drag and drop from the phonemes panel the phonemes which form the pronunciation of the current word into the current word panel. When the word is declared correct, the default&refine algorithm employs the phoneme-to-grapheme rule extraction method to extract new rules into the system’s rule sets for future use. As the words consist of the combinations of the graphemes (allowable letters in a language), grapheme-to-phoneme rule extraction method maps the phonemes with graphemes to create audible pronunciations. The mappings are obtained by using the iterative Viterbi algorithm as reflected in Appendix B (Viterbi, 1967).

During the rule extraction, each grapheme is given a phoneme (default) which is most likely to map to it. When the default phoneme is not correct, i.e., when the expected default phoneme is not correct, then potential graphemes that can map to the default phoneme are used and this is now extracted as a refined rule. The rules extracted are ordered in a reverse order, i.e., the last extracted rule becomes first one in the rule set and vice versa. Each rule is of the format

$$g_{left} - g - g_{right} \longrightarrow P \qquad \text{(Davel et al., 2008)}$$

Where the g is the grapheme being considered and g_{left} and g_{right} are the potential graphemes and P is the phoneme representing g.

Initially, the rule set is empty and the system keeps on extracting new rules and updating the rule set. With enough rule set extracted, the system starts to provide the correct and accurate predictions for words pronunciations. It is at this stage that the training is faster because the system is no longer updating the rule set each time a pronunciation is declared correct.

At the end of the dictionary development, the developer can view the project rule statistics (all rules in the project, rules per grapheme and rules per grapheme per context). In conclusion, Table 4.2 displays project rules statistics after the dictionary development.

Graphemes	Rules per grapheme	Graphemes	Rules per grapheme
g	5	o	5
h	15	p	9
i	2	r	11
j	5	s	13
k	4	t	2
l	5	u	2
m	9	w	12
n	1	y	5
x	12	b	2
a	5	e	5
d	3	h	14

Table 4.2: number of rules per grapheme in the rule set

5 An Automatic Speech Recognizer

In this chapter:

- *Components of Automatic speech Recognition*
- *Definition of Hidden Markov model*
- *Development of a speech recognizer using Hidden Markov Toolkit*
- *Problems encountered during the speech recognizer development*

5.1 Introduction

Speech is the primary mode of communication among people. When people speak, they *recognize* and *understand* the meaning behind words. Automatic speech recognition (ASR) refers to a process through which the computer can recognize the words spoken by a person.

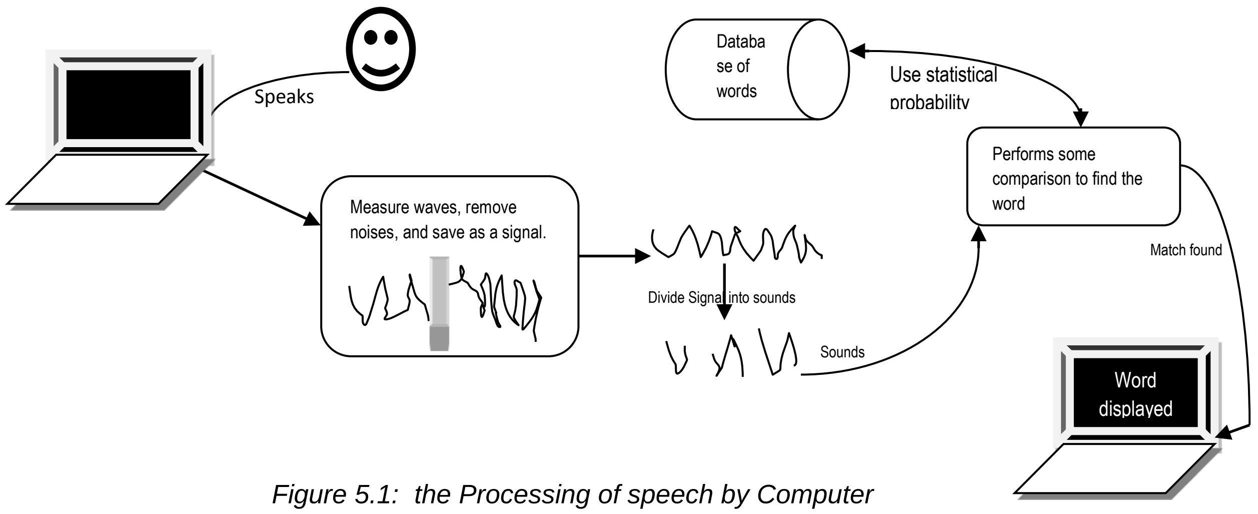

Speaks
Database of words
Use statistical probability
Performs some comparison to find the word
Measure waves, remove noises, and save as a signal.
Divide Signal into sounds
Sounds
Match found
Word displayed

Figure 5.1: the Processing of speech by Computer

When a person speaks, the computer "*listens".* It then removes or "conveniently ignores" the unnecessary noises from the speech. It measures the waves of the speech and stores these as a signal. The computer divides the signal into basic sounds that make up the words. It uses the statistical probability approach to find or predict the uttered words based on the digital signal representation within a specific context. It also uses a database of words in the computer for comparison. When it finds an appropriate match, it responds by displaying the answer on the monitor. Computers use automatic speech recognition systems to be able to transcribe human speech into text. Figure 5.1 shows how a computer processes speech. In this chapter we look at automatic speech recognition systems and the components they use to understand speech. We also look at the procedure used to develop a simple automatic speech recognizer for this study.

5.2 Components of automatic speech recognition

In general, there are three types of ASR systems: *speaker dependent*, *adaptive* and *speaker independent*. A speaker dependent system is designed to function for a single user. Such a system is easier to develop and is generally more accurate in recognition. However, the downside of it is that it is not flexible. An adaptive system is designed to adapt its functionality to the attributes of new speakers. A speaker independent system is designed to be used by any speaker.

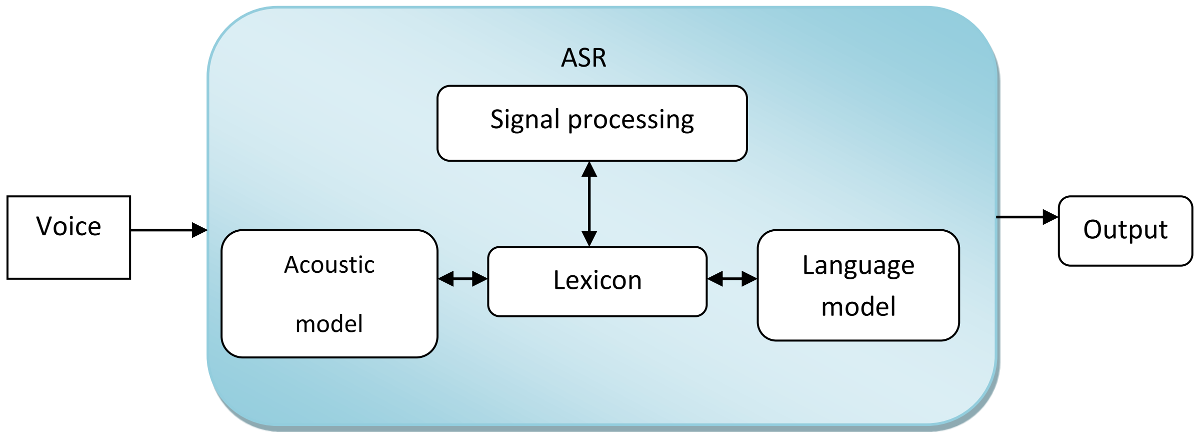


Figure 5.2: A simple ASR system

Most ASR systems use the statistical acoustic and language models to find the most probable word sequence, **Ŵ,** that has a maximum posterior probability **P(W|X)** when using the Bayes theory.

$$\hat{\mathbf{W}} = \text{argmax}_{w}\ \mathbf{P(W|X)}$$

$$= \text{argmax}_{w}\ \mathbf{P(X|W)P(W)/P(X)} \qquad (5.1)$$

where **X** represents the acoustic feature sequence. **P (W)** is the language model.

As depicted in Figure 5.2, during recognition, the system receives speech as an input. The signal processing component converts the speech into waveforms that the system itself can deal with easily. The acoustic model uses the waveforms to generate a sequence of symbols and then compares them to a set of words in the lexicon. The lexicon and the language model work hand in hand in dealing with the *restrictions* that are present in a language. As the lexicon component has been already discussed in the previous chapter, the other components, namely, signal processing, language model and the acoustic model will be looked at briefly.

5.2.1 Signal Processing

The signal prosessing component receives the raw speech input . It plays a vital role during recognition in that, it ensures that information contained in the speech signals is easily extracted by the system, removes noises from the speech input and also reduces the data rate. It analyzes and transforms signals to and from analog and digital format as required.

During recognition phase, firstly, the speech input is converted into a format that computers can process very well, namely, the digital format of zero’s and one’s. Besides computers “understanding” the digital format so well, the processing of digital signals provides high processing speed and accuracy. The signal processor analyzes these signals by representing and transforming them to ensure that the relevant identification information is extracted. The signals are represented as a sum of sinusoid by using Fourier Series (Huang et al., 2001).

The information extracted from a signal is utilized by the subsequent components of the ASR system for further processing. When the processing of all the subsequent components is completed and a relevant match has been found, the signal processing component converts the digital signals back to analog signals. The found match is then displayed on the computer monitor.

5.2.2 Language model

The language model is used during recognition to recognize speech. It consists of a list of words and the probabilty of their occurrence. This component works closely with the lexicon during recognition to *restrict search* by limiting the number of possible words that need to be considered at any point in the search. The language model assigns probability to a sequence of words by means of the probability distribution **P(W),** using the *n-grams*. It contains a set of rules for a language that is used as the primary context for recognizing words. It also captures regularities in the language. The complexity of the language is directly related to the size of the data on which it is trained. The acoustic model uses this component to check on the restrictions that are present in the language.

The grammars contained in the language model can be made of the formal linguistic rules or stochastic models. Formal linguistic rules can however, bring up computational demands when included in the ASR system. The n-grams, as one of the stochastic grammars, are easy to use, implement and interface with an ASR system and they are also good predictors of the short term dependencies (Meeter and Rohlick, 1993). The n-grams predict the $\mathbf{x_i}$ word sequence based on the $\mathbf{x_{i-1} \ldots, x_{i-n}}$ (Huang et al, 2001). In an n-gram model, the probability of **P(X)** over a word string $\mathbf{X = \{ x_1, x_2, \ldots, x_n \}}$ that shows how frequent a string **X** occurs as a sentence is given by $\mathbf{P(X) = P(x_1, x_2, \ldots, x_n)}$

$$= \mathbf{P(x_i)\, P(x_2|x_1)\, P(x_3|x_2, x_1) \ldots\ldots. P(x_n|x_1, x_2 \ldots x_{n-1})}$$

$$= \prod_{i=1}^{n} \mathbf{P\,(x_i|x_1, x_2 \ldots x_{i-1})} \qquad (5.2)$$

where the $\mathbf{P(x_i|x_1, x_2 \ldots x_{i-1})}$ is the probability that $\mathbf{x_i}$ will follow from $\mathbf{(x_1, x_2 \ldots x_{i-1})}$

When developing the language model, the training data from a specific domain together with a dictionary are used to create appropriate vocabulary. This ensures that the percentage of the out-of-vocabulary(OOV) word error rate(WER) is decreased. The WER affects the quality of the system and at times, can lead to less recognition accuracy. Also, it can make the system less flexible (Huang et al., 2001).

5.2.3 Acoustic Model

Acoustic modelling refers to the development of representations of the sounds that make up each word. These representations, called hidden Markov models (HMMs), are developed using the Hidden Markov Model Toolkit (HTK). Huang et al., (2001), define the HMM as "powerful statistical method of charecterizing an unobserved data samples of a discrete time series ". The HMMs are based on Markov chain. They further point out that the Markov chain is a sequence of random variables $X_1,X_2,X_3,\ldots$ with the Markov property given that the present state, the future and past states are nearly independent,.i.e.,

$$\mathbf{P\ (X_{n+1} = x \mid X_1 = x_1, X_2 = x_2,\ldots, X_n = x_n) = P(X_{n+1} = x \mid X_n = x_n)} \qquad (5.3)$$

The possible values of X_i form a countable set S called state space of the chain. The Markov chain moves through one state to another between a countable number of possible states. A simple two state Markov chain is shown in Figure 5.3.

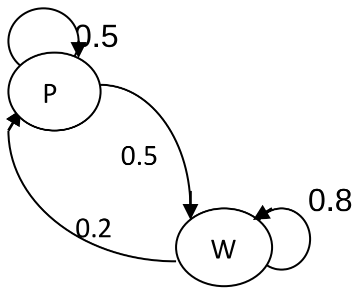


Figure 5.3: A two state Markov chain

By applying the Markov chain concept, Shokhirev (2010*)* defines a Hidden Markov Model formally as:

- Hidden states $\mathbf{Q} = \{ q_i \}$, $i = 1, \ldots, N$.
- Transition probabilities $\mathbf{A} = \{a_{ij} = P(q_j \text{ at } t+1 \mid q_i \text{ at } t)\}$, where $P(a \mid b)$ is the conditional probability of a given b, $t = 1, \ldots, T$ is time, and q_i in $\mathbf{Q}$.

Informally, ***A*** is the probability that the next state is q_j given that the current state is q_i.

- Observations symbols ***O*** = { o_k}, $k = 1, \dots, M$.

- Emission probabilities ***B*** = { $b_{ik} = b_i(o_k) = P(o_k \mid q_i)$ }, where o_k in ***O***. Informally, ***B*** is the probability that the output is o_k given that the current state is q_i.
- Initial state probabilities ***Π*** = {$p_i = P(q_i$ at $t = 1)$}.

Using this formal definition of the HMMs, Huang et al., (2001), further state that most applications of the HMMs solve the following problems:

Evaluation problem

When given model parameters, how to find the probability of a certain output sequence. The evaluation problem is resolved by making use of the forward algorithm (see Appendix C) which determines the probability of the partial observation sequence in an increasing length.

- Decoding problem
  Having the model parameters, what can be the most likely sequence of states that might have generated a given output sequence. This problem is resolved by applying the Virtebi algorithm (see Appendix A) which selects the best state sequence that maximizes the possibility of the state sequence for the given observation sequence.
- Learning problem
  Having the output sequence, what can be the most likely set of state transition and output probabilities. The learning problem is solved by using the Baum-Welch algorithm (see Appendix C) which is usually used to find unknown parameters of the HMM.

5.3 Hidden Markov Model Toolkit (HTK)

The HTK was used to build and manipulate HMMs (Young et al., 2002.). It is mostly used for speech recognition but, it has also been used in many other pattern recognition applications that employ HMMs. The HTK has been developed by the speech recognition group at Cambridge University’s Engineering Department. It is designed for building the HMM-based speech processing systems, such as an automatic speech recognition system. In this section, we describe the step-by-step procedure used to develop a simple Northern Sotho sub-word-based continuous speech recognizer. The system is developed in a Windows 7 environment with Visual Studio and Perl scripting language installed before installing the HTK.

The HTK toolkit works mainly with library modules. Figure 5.3 shows the HTK Software architecture. By fitting the ASR components shown by Figure 5.1 into this architecture, signal processing is handled by the HSigP library, HLM deals with language modelling, the HDict handles the lexicon and acoustic modelling is performed in the HModel library. Appendix D describes what t each of these libraries does.

HAudio	HLabel	HLM	HNet	HDict
Hwave	HTK Tool			HModel
HParm				HUnit
HVQ				
HSigP				HShell
HMem				HGraf
HMath	HTrain	HFB	HAdapt	HRec

Table 5.1: HTK software architecture (adopted from Young et al., 2002.)

5.3.1 Creating speech recognizer using HTK

This section provides a *summarized description of the step-by-step creation of the speech recognizer developed in this study.*

Firstly, Active Perl 5.12.3 was installed as there are some scripts that are used by the HTK during the training process. Visual Studio 6 (VC6) was also installed because HTK 3.4 cannot compile on Windows without Visual Studio. By installing the HTK, these executable files are automatically created and ready to be used:

HBuild, HCompP, HCopy, HDMan, HEAdapt, HERest, HHED, HInit, HLEd, HLStats, HList, HParse, HQuant, HRest, HResults, HSGen, HSLab, HSmooth and HVite. The data to be used for the training are prepared and then the monophones models which are later expanded to triphones are trained. The final HMMs are of continuous density mixture Gaussian tied state with clustering performed using decision trees.

Data preparation

- Task grammar

  The task grammar is the recognition grammar that defines what the recognizer can anticipate as an input. During recognition, when the recognizer "hears" a word or a phrase, it checks with the defined grammar and returns the grammar to the calling program. The grammar is compiled on a command prompt using the HParse command to create word level lattices files which are used at a later stage.
- The Pronunciation Dictionary

  All the words that are in the grammar should be in the dictionary. HTK Perl script, *prompt2wlist*, is used on the grammar to print each word on a line into a wordlist to be included in the dictionary. The wordlist file is incorporated with the dictionary created previously in chapter 4. HDMan command is executed to generate a complete pronunciation dictionary and a monophones file that lists all the phones used in the dictionary. A phonetically well-balanced and alphabetically ordered dictionary is created. Each word together with its corresponding phonemes that make up that word is on a single line. This can now be used to create a balanced acoustic model.
- Recording the data

  The training and testing data are recorded, preferably, in a noise free room. In our study, the Praat system was used to do all the required recordings. Firstly, the HSGen command is used on the word level lattices files created in step 1. This generates a script of sentences, called *sentence.txt*, that are to be recorded. Each sentence is recorded and stored in a waveform format in a file.

- Creating the transcription files
  For each sentence in the *sentence.txt* file, a single file called Master Label File (MLF) which contains a label entry is created. This is because HTK cannot process the *sentence.txt,* directly but rather uses the MLF to process the sentences. The HTK script, *prompts2mlf,* is used on the *sentence.txt* to generate the MLF file, *sentence.mlf*. The HLEd command is executed using the *sentence.mlf* to replace each word with its phonemes and the results are stored in a file named *phones.mlf*. The short-pause (sp) model is placed between the words to accommodate any pauses introduced by the speaker during recording, and this is stored in a file named *phones1.mlf*. *Phones.mlf* contains the required phone level transcriptions.
- Coding the Audio data
  The recorded speech waveforms are converted into feature vectors called Mel Frequency Cepstral Coefficients (MFCC). A script, *codetrain.scp*, which contains a list of source audio files and the names of the MFCC files to be converted to, is created. Conversion parameters are specified in a configuration file called *config.txt*. The HCopy command is executed to convert all the wave files to MFCCs in the codetrain.scp. The required data for the training of the HMM is now ready.

Creating Monophones HMMs

- Creating flat start monophones
  First of all, the prototype model, proto, and the configuration files are defined (see Appendix D). The script file, *train.scp*, which contains the paths of the MFCC files created previously, is created. HCompV command is executed to generate a new proto model in the folder, hmm0, which will be used to define HMMs for each phone. In order to create the flat start monophones, the monophones file generated previously in step 2 of the data preparation, is used as the HMM definition file (hmmdef). Each phone is put in double quotes preceded by '~h'. The newly created proto model is added to each phone in the hmmdef. The HEREst command is executed twice to re-estimate the HMMs in the hmm folder.

- Fixing the silence models

  The short-pause models are appended to the flat monophones. This is done by adding the *'sp'* model to the hmmdef file. The *'sp'* model is created by copying the centre state of the *sil* model in the hmmdef file. The HHED command is executed twice to tie this model to the *sil* centre state.
- Realigning the training data

  The HVite command is executed to realign the training data and to accommodate words that have more than one pronunciation. The best pronunciation that matches the acoustic data is used in a case where a word has more than one pronunciation. When executing this command, a HVite log is kept on track. The log is analyzed to ensure that each word in the *sentence.txt* is recognized and to catch errors. This helps in avoiding small errors that might cause some serious damage at the later stage. When everything is in order in the Hvite log, the HERest command is executed twice to re-estimate the HMMs in the hmm folder. Thus far, monophones HMMs are created.

Creating the tied-state triphones

- Triphones from monophones

  Monophones transcriptions created in the previous step are converted to triphones transcriptions. Triphones are a group of three phones formed from a single phone. Each phone is replaced with right neighbour phone, the main phone, and the left neighbour phone (e.g. L-X-R). The *MKtri.led* edit script is created (see appendix D). The HLEd command is executed to convert the monophones to triphones. The HMM models (hmmdef) are re-estimated by executing the HERest command twice.
- Tied-state triphones

  The *global.ded* script (see appendix D) is used together with the HDman command to run against the lexicon to generate a new version of the lexicon which consists of words with their pronunciations represented using triphones. The HHEd command is executed to perform decision tree state tying and the output is saved in a log file for the purpose of threshold tuning.

Finally, the re-estimation of the HMMs and creation of a tied list file is performed by running the HERest command twice. The tied list together with the HMMs can now be used to recognize speech. Figure 5.3 summarizes all the HTK processing stages for the development of a speech recognizer using HTK toolkit.

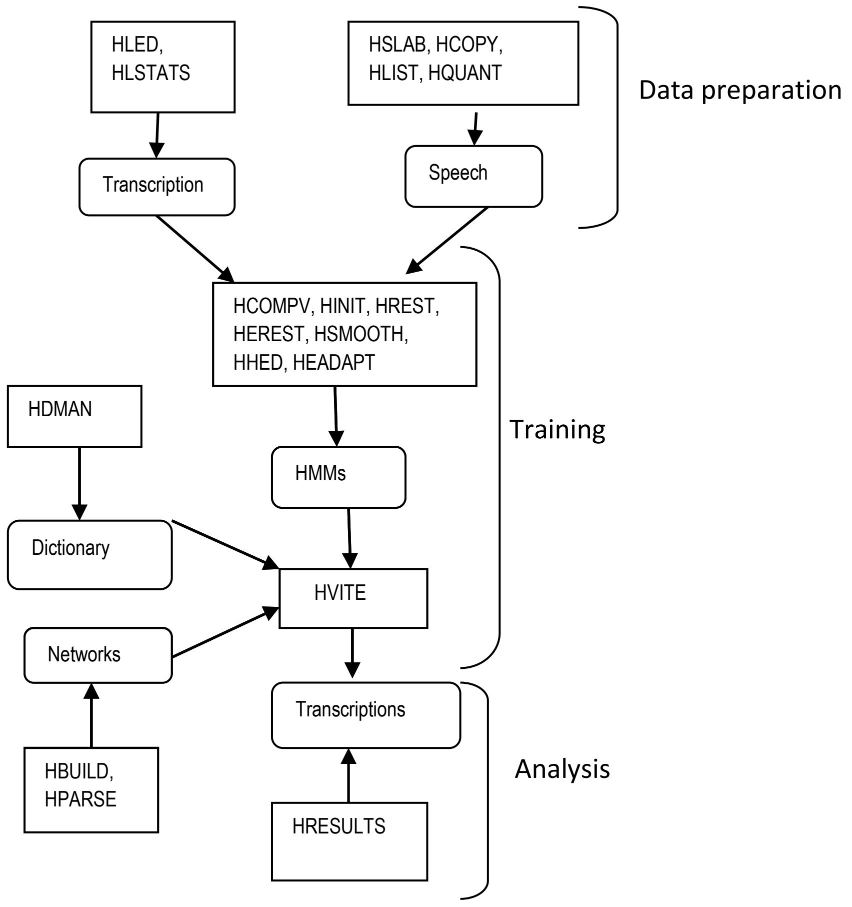


Figure 5.4: The HTK system processing stages (adopted from Young et al., 2002)

5.3.2 Problems encountered during the development process

Before creating the triphones model from the step-by-step process discussed earlier, speech recognition was performed using the monophones model. The accuracy was much higher as compared to when using the triphones model. The reason for this is attributed to the triphones model representing a given phoneme with a particular right hand phoneme and a particular left hand phoneme. According to Precoda k., (2004), this causes the challenge of number of the indicated phonemes to be very large. Many examples of each triphone had to be collected because triphones do not capture all the relative factors influencing a phonetic realization. The large number of phonetic realization led to thousands of different utterances requiring an expensive data recording effort.

6 Results and discussion

In this chapter:

- *Introduction*
- *ASR with a smaller sized dictionary*
- *ASR with an increased size dictionary*
- *ASR with an increased training speech corpus*
- *The Overall performance of the ASR system*
- *Extraction of rules by the Dictmaker*

6.1 Introduction

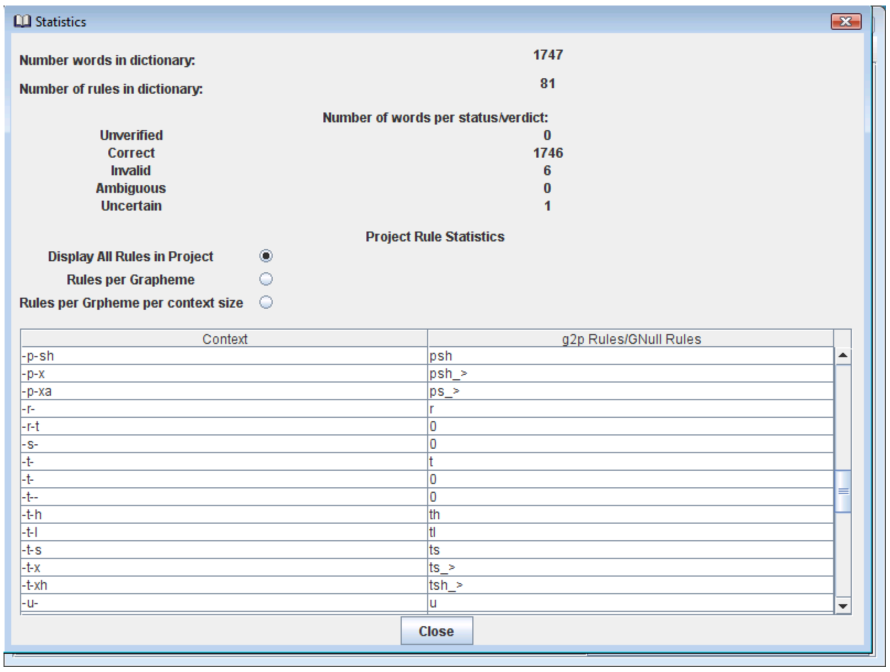
The aim of this study is to develop a more comprehensive and representative pronunciation dictionary that can be incorporated into a speech recognition system. In chapter 4, the discussion centered around the method and approach used to develop the pronunciation dictionary. In the previous chapter, the implementation of the ASR system was discussed. This chapter is dedicated to the experimental results achieved in evaluating the accuracy of the methods used to create the dictionary and to measure the performance of the ASR system. This chapter begins by discussing the results obtained from using the Dictmaker in the creation of the dictionary. The dictionary was initially created with a smaller sized wordlist and it was then incorporated into the Northern Sotho ASR system. At a later stage, the dictionary size was increased and the dictionary was again incorporated into the system. In both the instances, the performance of the ASR system was monitored. The performance of the ASR system was also monitored when the size of the training data of the ASR was increased. Lastly, the statistics of the extraction of the rule set by the default&refine algorithm of the Dictmaker are discussed.

6.2 Experiment I: ASR with a smaller sized dictionary

*6.2.*1 Morphological approach to the creation of the dictionary using Dictmaker tool

A phoneme set consisting of 42 phonemes in the Northern Sotho was initialized to Dictmaker for it to be able to provide the pronunciations of the words in the wordlist. One speaker was used to record all these phonemes. The recordings were done in a quiet room using the Praat program at a sampling frequency of 44100 HZ.

Using the syllable structures discussed in chapter 3, a verb list consisting of 200 Northern Sotho verbs was systematically created. Most of the verbs in verb list were used because they can be easily modified to express different grammatical categories when affixes are appended to them. Using the morphological approach discussed in chapter 3, a Perl program which analyzed each of these verbs to generate new valid word forms was developed. From each verb, the program should generate *ten new valid* word forms. However, due to some restrictions of some meanings and the semantic correctness, the program was unable to generate new words from some of the verbs, such as "*hloko*" (take note/be careful).


Statistics

Number words in dictionary:	1747
Number of rules in dictionary:	81

Number of words per status/verdict:

Unverified	0
Correct	1746
Invalid	6
Ambiguous	0
Uncertain	1

Project Rule Statistics

[✓ Display All Rules in Project]
Rules per Grapheme
Rules per Grpheme per context size

Context	g2p Rules/GNull Rules
-p-sh	psh
-p-x	psh_>
-p-xa	ps_>
-r-	r
-r-t	0
-s-	0
-t-	t
-t-	0
-t--	0
-t-h	th
-t-l	tl
-t-s	ts
-t-x	ts_>
-t-xh	tsh_>
-u-	u

Figure 6.1: Dictmaker project analysis of the smaller sized dictionary

In total, from 200 verbs, the program was able to generate 1747 new valid word forms in Northern Sotho. The new valid word forms together with the verb list were used as the wordlist for the dictionary training. From Figure 6.1, the dictionary was initialized with 1753 words. During training, 6 words were declared invalid due to syntactic correctness, and therefore 1747 words were used for the training. In total, eighty-one rules were extracted and used for pronunciations using the default and refine algorithm. All words were verified, 1746 were correctly pronounced, and these words were included in the final dictionary. One word was declared uncertain and as result was excluded in the final dictionary. The bottom part of Figure 6.1 shows the statistics of the rules extracted in the project. The words with graphemes listed in the context column were replaced with the corresponding phonemes in the g2p (grapheme-to-phoneme) rules column. Overall, 99.94% of accuracy of the final dictionary was obtained by using this formula:

Dictionary accuracy = 100% * *(number of correctly pronounced words in the dictionary/ number of the words in the training dictionary).*

6.2.2 ASR system in Northern Sotho using HTK

Two speakers were used to record the speech data required for the training and the testing of the ASR system. In total, 260 sentences were recorded in a relatively quiet room using the Praat system at the sampling frequency of 44100 HZ. One speaker recorded 130 sentences for the training and the other also recorded the same number of sentences for the testing. The sentences contained verbs, and the domain of these sentences could not be classified. The hidden Markov models were trained and the dictionary created was incorporated with this speech recognizer. The HMMs were of continuous density Gaussian mixture with clustering done using decision trees. The HTK tool, HResults, was used to evaluate and test the recognizer.

```
------------------------ HTK Results Analysis----------------------------------------------

         SENT: %Correct=38.09 [H=35, S=91, N=130]

          WORD: %Corr=60.53, Acc=58.39 [H=158, D=3, S=283, I=216, N=1177
      =======================================================
```

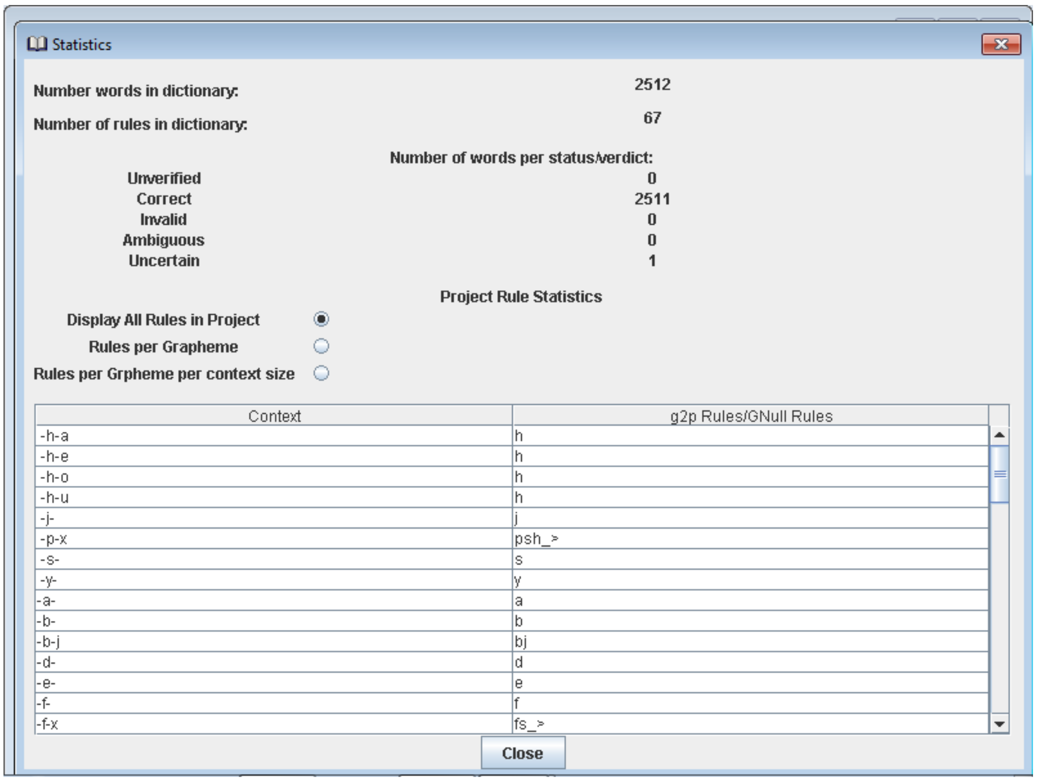
Table 6.1: HTK results analysis of the HTK-based speech recognizer with the small sized dictionary

From Table 6.1, out of 130 sentences, 50 (38.09%) were correctly recognized. Out of 1177 words, 712 (60.53%) were correctly recognized. There were 3 deletion errors (D), 283 substitution errors(S), and 216 insertion errors (I). Accuracy (Acc = 58.39) percentage is lesser than the correctness (Corr = 60.53) percentage because the accuracy considers the insertion errors.

6.3 Experiment II: ASR with an increased size of the dictionary

6.3.1 Morphological approach to the creation of the dictionary using Dictmaker tool

The same phoneme set from experiment I was used. The verb list had 300 verbs in Northern Sotho. The Perl program generated 2512 new valid word forms from the verb list. The wordlist consisting of 2512 words was initialized to Dictmaker.


Statistics

Number words in dictionary:	2512
Number of rules in dictionary:	67

Number of words per status/verdict:

Unverified	0
Correct	2511
Invalid	0
Ambiguous	0
Uncertain	1

Project Rule Statistics

Display All Rules in Project [✓]
Rules per Grapheme
Rules per Grpheme per context size

Context	g2p Rules/GNull Rules
-h-a	h
-h-e	h
-h-o	h
-h-u	h
-j-	j
-p-x	psh_>
-s-	s
-y-	y
-a-	a
-b-	b
-b-j	bj
-d-	d
-e-	e
-f-	f
-f-x	fs_>

Figure 6.2: Dictmaker project analysis of the increased sized dictionary

From Figure 6.2, 2511 words were declared correct and were included in the final dictionary. One word was declared uncertain. Sixty-seven rules were extracted and 99.96 percentage of accuracy of the dictionary was obtained.

6.3.2 ASR system in Northern Sotho using HTK

The improved dictionary was incorporated into a simple ASR system. The same training and testing data from experiment I was used.

```
------------------------ HTK Results Analysis-----------------------------------------------
          SENT: %Correct=30.00 [H=39, S=90, N=130]
          WORD: %Corr=63.91, Acc=59.40 [H=920, D=3, S=255, I=214, N=1177
============================================================
```

Table 6.2: HTK results analysis of the HTK-based speech recognizer with an improved dictionary

In total, out of 130 sentences, 39 (30%) were correctly recognized. Out of 1177 words, 752 (63.9%) were correctly recognized. There were 3 deletion errors (D), 255 substitution errors(S), and 214 insertion errors (I). As usual, the Accuracy (Acc = 59.40) percentage was lesser than the correctness (Corr = 63.91) percentage.

6.4 Experiment III: ASR with an increased size of the training data

To improve the performance of the ASR system, the speech data for the training of the speech recognizer was increased. Additional 70 sentences were added to the speech data. In total, 400 sentences were recorded. Two hundred sentences were for the training and the other 200 sentences were for testing. Table 6.3 depicts the results obtained from using the HTK HResults.

```
---------------------------------- HTK Results Analysis------------------------------------------------
          SENT: %Correct=44.00 [H=88, S=112, N=200]
          WORD: %Corr=82.07, Acc=63.15 [H=755, D=10, S=155, I=174, N=920]
=================================================================
```

Table 6.3: HTK results analysis of the HTK-based speech recognizer with an increased corpus

In total, out of 200 sentences, 88(44%) were correctly recognized. Out of 920 words, 755 (82.07) were correctly recognized. There were 10 deletion errors (D), 755 substitution errors(S), and 155 insertion errors (I). The Accuracy (Acc = 63.15) percentage was lesser than the correctness (Corr = 82.07) percentage.

6.5 The Overall Results of the ASR System

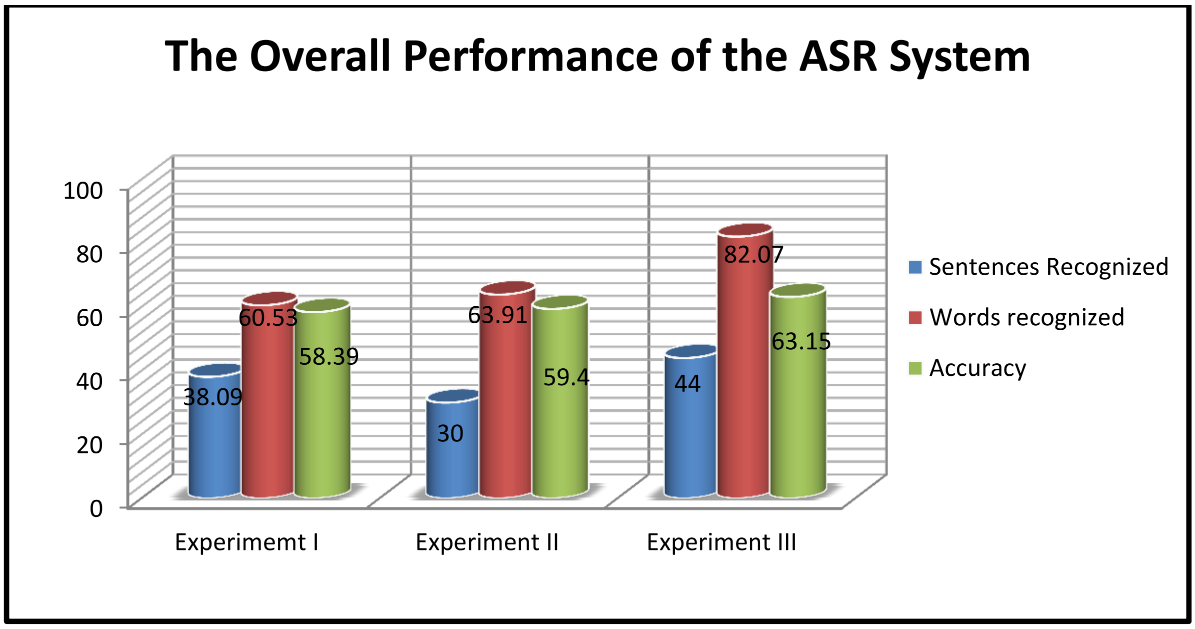


Figure 6.3: The overall performance of the ASR Systems

When comparing experiment I with experiment II in Figure 6.3, the accuracy percentage improved by 1% (from 58.39 to 59.40). This was due to the decrease in number of the insertion errors. However, the sentence recognition accuracy percentage dropped by 8.09% (from 38% to 30%). In Experiment III, there was a significant improvement in the word recognition rate and a slight improvement in both the system accuracy and the sentence recognition rate.

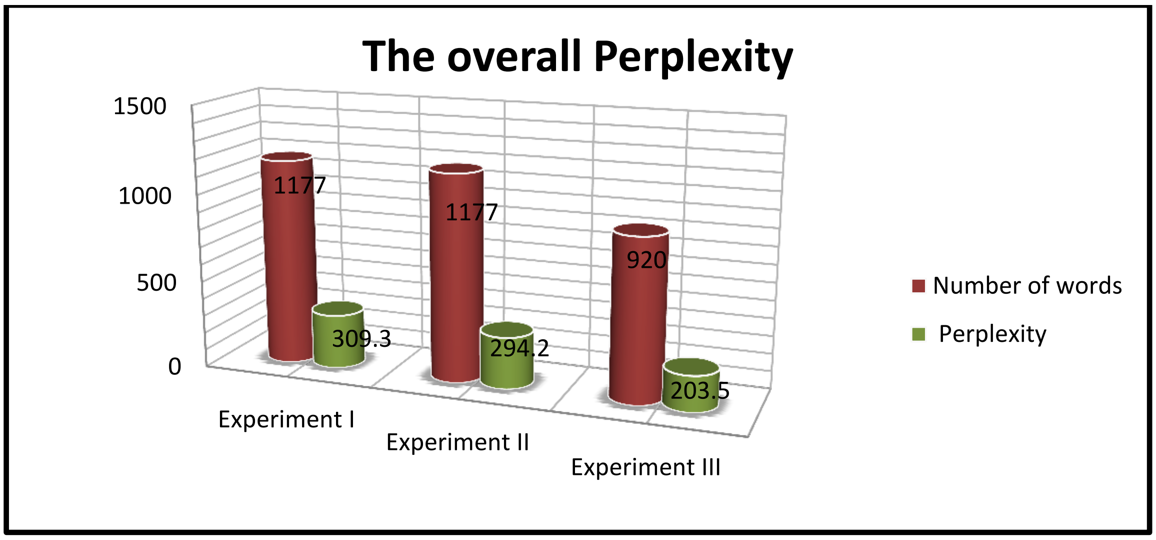


Figure 6.4: The overall perplexity of the ASR Systems

The perplexity of each trigram language model from each experiment is shown in Figure 6.4. The perplexity was based on the mean probability of the words in each experiment. The language model in experiment III has outperformed the other two as it has the lowest perplexity. This might be due to the lesser number of words in Experiment III.

6.6 The Extraction of rules by the Default&Refine Algorithm

Figure 6.1 and Figure 6.2 showed the dictmaker project analysis in which the statistics of the number of rules that were extracted during the dictionary development were also shown. The bottom part of these two figures showed as to which graphemes were replaced with what phonemes during the training. This section looks at the rate at which these rules were extracted.

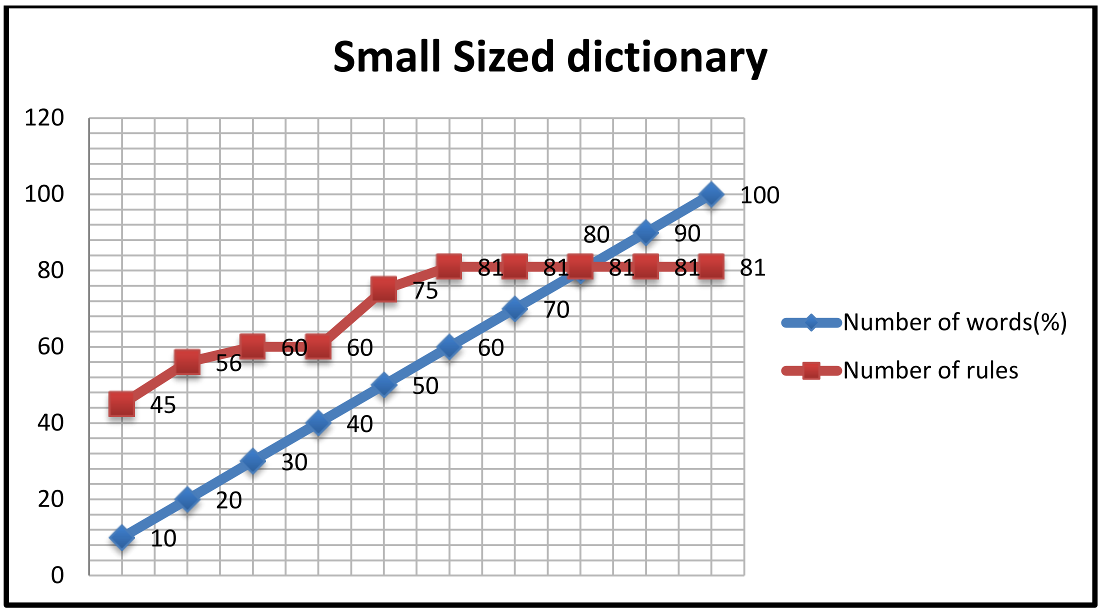


Figure 6.5: Small sized dictionary: Number of rules vs. Number of words

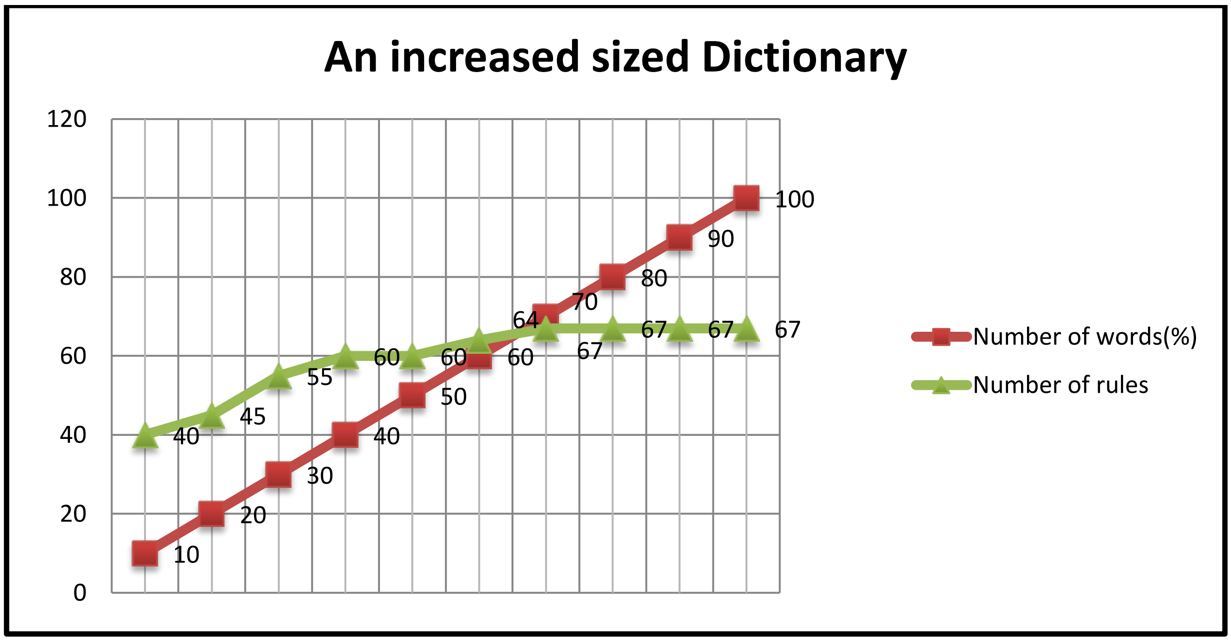


Figure 6.6: The improved dictionary: Number of rules vs. Number of words

The Dictmaker extracts as many rules as possible at the very beginning of the training. From Figure 6.5 and Figure 6.6, when 10% of the training words were declared correct by the developer, 45 and 40 rules respectively, were already extracted to the system. Between 30% and 40% of the training words in the small sized dictionary, and between 40% and 50% of the training words in the improved dictionary, the number of the extracted rules was constant. It was at this stage that there were no new rules extracted to the rule set and the Dictmaker was using the already extracted rules to predict accurate pronunciations. Between the 60% and 100% number of the training words in the small sized dictionary, and between the 70% and 100% of the training words in the improved dictionary, the number of rules did not change, 81 and 67 respectively. This means that the Dictmaker had all the required rules in the rule set and as a result, it was able to provide the correct pronunciations without the developer replacing any of the predicted phonemes. This made training much faster because the more the rules in the rule set the more accurate the Dictmaker provided the correct predictions.

In total, out of the 1747 words in the small sized dictionary, 40% (698) of the words were correctly pronounced by the Dictmaker. Out of the 2511 words in the improved dictionary, 30% (753) of the words were correctly pronounced by the Dictmaker. However, these higher percentages of the words correctly pronounced by the Dictmaker were achieved due to the wordlists having only the verbs and the new valid word forms which were formed by appending ten affixes to different verbs. The new valid word forms were inflectionally related and this made the extraction of the rules much easier.

7 Discussions, Conclusion and Future Work

In this chapter:

- *Concluding discussion*
- *The overall conclusion*
- *The future work*

7.1 Introduction

The main task of this study was to develop a more comprehensive and representative pronunciation dictionary in Northern Sotho. The pronunciation dictionaries are some of the important requirements in developing speech and language technology systems, such as ASR, TTS and translation systems. The speech and language technology is currently trying to improve people's interaction with computers and related computational devices by using speech interface, which are proving to be the popular with people using a simple and widely used device (mobile) to access information, to use email and even to do online transactions in their home languages.

In this research report, chapter one discussed the background on speech and language technology and the literature review on ASR. Chapter two looked at the basic verbs in Northern Sotho which were the main input data for the creation of a pronunciation dictionary. Chapter three discussed the morphological and rule-based approach which was used in developing the pronunciation dictionary. Chapter four looked at the methodology and tools used to create the pronunciation dictionary. Chapter five described the components of a typical ASR system, the tools used, and the manner in which a simple HTK-based ASR system was implemented.

Chapter six evaluated the methods and approaches used in developing the dictionary and incorporating it into an ASR system.

7.2 Discussions

Very recently, it has been discovered that speech and language technology (SLT) can become one of the solutions in closing digital divide that exists mostly in the rural areas of developing countries (Furuholt and Kristian, 2007). Most of these countries have a higher rate of illiteracy and as a result, its citizens cannot use computers or access valuable information. While on the other hand, the SLT is seen as a solution in accommodating the handicapped (blind, hard-of-hearing) community to whom oral communication is the main form of communication (Rudnicky et al., 1993). The SLT expands the availability and increases the effective usability of information communication technology (ICT). ASR technology as one of the SLT brings about the man-machine interaction in natural language using human voice. However, the use of this technology causes a significant challenge with specific reference to linguistic resources for the under-resourced languages spoken in developing countries.

Northern Sotho is one of the South African official languages which is regarded as under-resourced. Since there is not much computational research work done in this regard on marginalized and under- resourced, it is therefore very necessary to initiate more research work on the creation of lexicons - as one of the major components of the Northern Sotho ASR. The Dictmaker tool designed by the CSIR Meraka Institute was used to implement the lexicon. The tool was used because it is free and it can be used by the linguistically inexperienced dictionary developers to create accurate dictionaries at a faster pace.

The usage of the syllable structures in the formation of the verbs helped in generating verbs with good and diverse root structures. In Northern Sotho, a verb root states a verb's meaning whereas a verb root and an inflectional ending are used to state the unique meaning of the verb. This was the main reason for the usage of the verbs as the input to the creation of the dictionary and that inflectional ending can be used on a single verb to express many different meanings.

To evaluate the performance of the resultant lexicon in practice, a simple ASR system was developed using the HTK toolkit. There are many other free speech recognition engines such as SPHINX, Lumenvox, ISIP, etc. that are available for use but the HTK was preferred because it has been used mostly by many other SLT researchers. It also supports both the sub word based recognition and the isolated whole-word recognition. This study focused on the sub-word recognition as it can produce a sequence of sub-word units in place of OOV words. The recognizer was tested with recorded (stored) data and better recognition performance was achieved.

7.3 Conclusion

The main objectives of this study were to investigate the contributions of the morphological approach to generating a more comprehensive pronunciation dictionary in Northern Sotho. Another objective was to create an expanded pronunciation dictionary that can improve language coverage with respect to inflected words, thereby increasing the chances of a larger and more representative dictionary that has the potential to improve the performance of the ASR systems. In order to meet these objectives, a base verb list was systematically developed.

A morphological rule-based Perl program was constructed to analyze the base verb list and generate new valid inflected word forms by appending appropriate affixes to these verbs. The morphological rule-based program was able to generate valid word forms which were not commonly available in the existing Northern Sotho dictionary. Overall, 99.96% recognition accuracy was achieved in the final pronunciation dictionary created using the Dictmaker tool. The larger dictionary was used in developing an ASR system and the performance of the ASR system improved by one percent as compared to the initial system with a smaller sized dictionary. This implies that a larger dictionary does have a significant effect in the overall performance of the ASR system because it improves the language coverage of the system. Much better recognition accuracy was also achieved by improving the pronunciation dictionary in the ASR system.

The morphological approach to the development of the pronunciation dictionary has a significant impact in creating a larger and representative dictionary that can augment the existing methods of creating pronunciation dictionaries for the under-resourced African languages such as Northern Sotho. Once again, the performance of the simple HTK-based ASR system was slightly improved by increasing the training data for the building of the statistical models.

7.4 Future work

The ASR system was also to be tested with live data. However, the live testing process met with many implementation problems. During the live testing, the system could not “hear” the sounds (words) uttered for recognition. One of the reasons for this is that the HTK’s audio recorder tool, HSLab, couldn’t work during the data recording for the system’s training and this was also the main reason for the use of the Praat system for the recordings of data for the systems training. To evaluate the system with live data, a high performance two-pass large vocabulary continuous speech recognition decoder software called Julius, will have to be installed to work with the HTK toolkit. Cygwin will be installed as Julius’ main operating platform is Linux and other UNIX platforms (Julius, 2001). Audacity sound recorder will be used as the main audio recorder. The HTK will have to be reinstalled and everything else regarding testing will be redone.

As the morphological approach has proved to have a significant impact in generating large dictionaries, for future work, it is intended to apply this approach on both the *nouns* and *verbs* in Northern Sotho so as to have a larger and complete pronunciation dictionary. For the best recognition accuracy and performance, HMMs that provide an efficient way to building close parametric models would have to be used. More training data with regard to building the statistical language model will equally have to be used.

I. REFERENCES

Biadsy F, Habash N, and Hirschberg J., "*Improving the Arabic Pronunciation dictionary for Phone and Word Recognition with Linguistically-Based Pronunciation rules*", In the Proceedings of the North American Association for Computational Linguistics, Boulder, Colorado, USA, 2009, pp.397-405.

Census 2001, from: http://www.southafrica.info/about/people/population.htm

retrieved on 15 June 2011.

Chelba C., Bacchian M., and Schalkwyk J., (2010), Challenges in Automatic Speech Recognition: 2010-2020 -a decade visions from Academia and industry [online]. Available at: http://static.googleusercontent.com/external_content/untrusted_dlcp/research.google.com/en//pubs/archive/36913.pdf *,* retrieved on 23 Sept 2011

Davel M. and Barnard E., (2008). Pronunciation prediction with Default&Refine, *Computer Speech and language*, vol **22**, pp.374-393,.

Davel M. and Martirisian O., (2009). "*Pronunciation dictionary development in resource scarce environments*". In P*roceedings of Interspeech 2009,* Brighton, UK, pp. 2851-2854.

Developmemt Management Group Consulting LCC, (2010). "*Using Voice self-service to enhance the retail banking experience",* technical report*,* available at: http://improvecustomerexperience.co.uk/wp-content/uploads/DMG-Retail-Banking-Using-Voice-Self-Service.pdf, retrieved on 3 January 2012.

Furuholt B. and Kristian S., (2007). "*A rural-urban Digital divide? Regional aspects of Internet use in Tanzania*", In the Proceedings of the 9th International Conference on Social Implications of computers in Developing countries, Sao Paulo, Brazil, May 2007.

Huang H., Acero A. and Hon H., (2001). Spoken Language Processing, Prentice Hall, Upper Suddle River, N.J. pp 201- 257.

Joffe D., (2011), Sesotho Sa Leboa, [online], available at: http://www.africanlanguages.com/northern_sotho/ , retrieved Sept 2011.

Juang B.H. and Rabiner L.R., (2005). Automatic speech recognition- A brief History of Technology.Technical Report, available at: http://my.fit.edu/~vkepuska/ece5526/ASRHistory-Juang+Rabiner.pdf, retrieved on 10 December 2011.

Julius, (2001). Multipurpose large vocabulary continuous speech recognition engine. Technical report, Kyoto University. Translated from the original Julius-3.2-book by Ian Lane? Kyoto University.

Kruger, C.J. H., 2006. *Introduction to the morphology of Setswana*. Lincom Europe, Munchen, Germany.

Meeter M. and Rohlick J. R, (1993), “Statistical Language Modelling Combining N-gram and Context Free Grammars”, In Proc. of International Conference on Acoustic, Speech and Signal Processing., Vol. II, pp.37-40.

Mengjie, Z., (2001). *Overview of speech recognition and related machine learning techniques*, technical report. Available at: http://www.mcs.vuw.ac.nz/comp/Publications/archive/CS-TR-01/CS-TR-01-15.pdf, retrieved on December 10, 2011

Nuance Communications, 2011. *Reduce costs*, Technical Report, available at : http://www.nuance.com/for-healthcare/by-goals/reduce-costs/index.htm, retrieved on 3 January 2012.

Osterdorf M., and Bulyko I., (2002). The impact of Speech Recognition on Speech synthesis. Retrieved October 2011 from http://citeseer.ist.psu.edu/542390.html

Poulos G., and Lawrence J., (1994). A Linguistic Analysis of Northern Sotho, GoodWood, 1994, pp.115-244

Precoda K., (2004), Non-mainstream language and speech recognition: some challenges. *Calico Journal*, Vol **21**(2), pp 229-243.

Price C.S and Gee Q.H., (1988). Syllable Analysis of North Sotho and its Effect on Computerized Hyphenation. *South African Journal of Science*, Vol. **84**, pp.480-482.

Rudnicky, A.I., Lee, K.F., and Hauptmann, A.G. (1992), Survey of current speech technology. *Communications of the ACM*, **37** (3), pp.52-57.

Shokhirev N., (2010), Hidden Markov Models, [online], available at: http://www.shokhirev.com/nikolai/abc/alg/hmm/hmm.html, retrieved on Dec 2011.

Sunitha K.V.N, and Kalyani N., (2009). Improving word coverage using unsupervised morphological analyser, *Sadhana,* Vol. **34**, 5, pp.703-715

Van Lieshout P, (2004) "P*raat short tutorial",* University of Toronto, Graduate Department of speech-language pathology, Faculty of Medicine, Oral dynamics lab

Viterbi A, (1967), "Error bounds for convulutional codes and a asymptotically optimum decoding algorithm". IEEE Transactions on Information Theory 13, pp. 260-267

Wilpon, J. G., and Rabiner, L., R., (1979). Considerations in applying clustering techniques to speaker-independent word recognition. *Journal of Acoustic Society of America*. **66** (3), pp.663-673

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. *The HTK Book*. Cambridge University Engineering Department, Cambridgeshire, UK, for HTK version 3.4 edition, December 2006

Zue V.W., and Lamel L.F., (1986). An Expert Spectrogram Reader: A knowledge-based approach to speech recognition, In Proceedings Acoustic, Speech, and Signal Processing, IEE International conference, pp.1197-1200

II. APPENDICES

Appendix A: Northern Sotho Consonants

- b
- bj
- d
- f
- fs
- fš
- g
- h
- j
- k
- kg
- kh
- l
- m
- n
- p
- ph
- pš
- ps
- psh
- pšh
- r
- s
- t
- th
- tl
- tlh
- ts
- tsh
- tš
- tšh
- w
- y
- š

Appendix B: Viterbi algorithm

Suppose we are given states Y, initial probabilities π_i of being in state *i and transition probabilities* $a_{i,j}$ of transition from state *i to state* j. When we observe outputs $x_0,\ldots, x_t$. The sequence of $y_0 \ldots, y_t$ most likely to have produced the observations is given by the recurrence relations:

$V_{0,k} = P(x_0 | k) . \pi_k$

$V_{t,k} = P(x_t | k). \max_{y\in Y} (a_{y,k} V_{t-1,y})$

Where $V_{t,k}$ is the probability of the most probable state sequence responsible for the first t + 1 observation that has k as its final state. The path can be retrieved by saving the back pointers that recall which state y was used in the second equation. Let Ptr (k, t) be the function that returns the value of y used to compute $V_{t,k}$ if t > 0, or k if t = 0. Then

$y_t = \text{argmax}_{y\in Y} (V_{T,y})$

$y_{t-1} = \text{Ptr}(y_t, t)$

Appendix C: The Forward and Baum-Welch algorithms

The Forward Algorithm

Step 1: Initialization

$\alpha_1(i) = \pi_i b_i(X_1)$ $\quad 1 \leq i \leq N$

Step 2 : induction

$\alpha_t(j) = [\sum_{i=1}^{N}(\alpha_{\tau-1}(i)a_{ij})]\, b_j(X_t)$ $\quad 2 \leq t \leq T ; 1 \leq j \leq N$

Step 3: Termination

$P(X|\phi) = \sum_{i=1}^{N}(\alpha_T(i))$ if it is required to end in the final state, $P(X|f) = \alpha_T(_{SF})$

The Baum-Welch Algorithm

Step 1: initialization: choose an initial estimate ϕ .

Step 2: E-step: Compute auxiliary function $\mathbf{Q}\ (\phi,\ \hat{f})$ based on ϕ .

Step 3: M-step: Compute $\hat{f}$

Step 4: iteration: set $\phi = \hat{f}$, repeat from step 2 until convergence

Appendix D: HTk modules and scripts

1. HTK Library Modules

HShell – controls the user input/output and the interaction with the operating system

HMEM- controls all the memory management required during processing

HMAth- provides the math support

HSigP – provides all signal processing operations needed for speech analysis

HLabel – provides the interface for label file

HLM -provides the interface for language model files.

HNET – interface for networks and lattices

HDict – an interface for dictionaries

HVQ – provides interface for VQ codebooks

Hmodel – provides HMM definitions

HParm – provides the parameterization for all the speech input/output waveforms

HWave – provides the interface for all speech input and output waveform

HAudio – provides the interface for direct audio input

HGraf – provides simple interactive graphics

Hutil – provides a number of utility routines for manipulating HMMs

HTrain and HFB – contains support for the various HTK training tools.

HAdapt – provides support for various HTK adaptation tools

HREC – contains main recognition processing functions

2. Prototype (proto) Model

```
~o <VecSize> 25 <MFCC_0_D_N_Z>
~h "proto"
<BeginHMM>
 <NumStates> 5
 <State> 2
  <Mean> 25
   0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
  <Variance> 25
   1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
<State> 3
  <Mean> 25
   0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
  <Variance> 25
   1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
<State> 4
  <Mean> 25
   0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
  <Variance> 25
   1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 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>
```

3. Configuration file

```
TARGETKIND = MFCC_0_D_N_Z
TARGETRATE = 100000.0
SAVECOMPRESSED = T
SAVEWITHCRC = T
WINDOWSIZE = 250000.0
USEHAMMING = T
PREEMCOEF = 0.97
NUMCHANS = 26
CEPLIFTER = 22
NUMCEPS = 12
```

4. MKtri.led edit script

```
WB sp
WB sil
TC
```

5. Global.ded script

```
AS sp
RS cmu
MP sil sil sp
```

Appendix E: papers submitted at conferences

1. Paper presented at the Southern Africa Telecommunication Networks and Applications Conference (SATNAC) 2011 at the East London Convention Centre, Eastern Cape, South Africa, 4-7 Sept 2011

   Title: Creating a Pronunciation Dictionary for Automatic Speech Recognition- A Morphological Approach

   Authors: M.C. Nkosi, M.J.D. Manamela and N. Gasela
2. Paper presented at the University of Limpopo Faculty of Science and Agriculture Postgraduate day (FSA-PG) 2011 at Bolivia Lodge in Polokwane, Limpopo, South Africa, 30 September 2011.

   Title: Towards a Comprehensive Pronunciation Dictionary for Automatic Speech Recognition Using Morphological Approach

   Authors: M.C. Nkosi, M.J.D. Manamela, and N. Gasela

Creating a Pronunciation Dictionary for Automatic Speech Recognition - a Morphological approach

MC Nkosi[*], MJD Manamela and N Gasela

Department of Computer Science,

University of Limpopo (Turfloop Campus),

Private Bag X1106,

Sovenga, 0727

Tel: +27 15 268 2243/ +27 83 392 4327, Fax: +27 15 268 3183

e- mail:mphocaselina@ymail.com; {200305303, jonas.manamela, nalson.gasela@ul.ac.za}

ABSTRACT- The Pronunciation dictionaries or lexicons play an important role in guiding the predictive powers of the Automatic Speech Recognition (ASR). As the use of automatic speech recognition systems increases, there is a need for the development of dictionaries that cover a large number of inflected word forms to enhance the performance of ASR systems. This paper describes the morphological-driven approach to the creation of a more comprehensive and broadly representative Northern Sotho pronunciation dictionary for ASR systems to augment the existing data-driven methods of dictionary creation. The Preliminary results of an ASR system using morphologically–based rules for a dictionary creation are discussed.

Index Terms- Automatic Speech Recognition, Northern Sotho, Pronunciation Dictionary, lexicon, hidden Markov model, Dictmaker.

I. INTRODUCTION

Automatic speech recognition technology has recently gained popularity in the society, especially in the customer services companies. A well-trained basic ASR system can recognize single word entries such as yes/no responses and spoken words. This makes it easier for people to make use of the automatic menus rather than having to use keypad to enter a bunch of numbers without any control of errors. The use of speech technology systems in the modern society is not only increasing, but, it has also become a necessary human-computer interface development of language engineering. An ASR system relies heavily on the comprehensiveness of its lexicon – a mapping of words to their corresponding pronunciation forms in terms of the phonemes/allophones in a specific natural language. As the use of the ASR systems in spoken language processing is increasing, there is a great need for the development of lexicons that cover a large number of inflected words for the under-resourced official languages of South Africa. The South African indigenous languages are regarded as the morphologically rich because of the large number of distinct word forms, each of which has a large number of possible pronunciations [1]. For such languages, it is difficult to create a comprehensive pronunciation dictionary by hand [2].

This research paper discusses the creation of a lexicon using the morphological approach on the verb forms in Northern Sotho – one of the South Africa's official languages. Northern Sotho is predominantly spoken in the Limpopo Province of the South Africa by at least 9% of the total RSA population [3]. It belongs to the Sotho cluster of the African indigenous languages sharing more common features with Sesotho and Setswana.

Section 2 reviews a general speech recognition system. Section 3 outlines the morphological approach to the development of the rules for the pronunciation dictionary creation. Section 4 describes the process of creating the pronunciation dictionary. The preliminary results of the ASR system are presented and discussed in Section 5 before giving concluding remarks in Section 6.

II. AUTOMATIC SPEECH RECOGNITION OVERVIEW

Speech is the primary means of communication among people. When people speak to each other, they make utterances to one another, listen and recognize and understand the meaning behind the words. Automatic speech recognition is a process through which a computer system can recognize the words spoken by a person and coverts them to text.

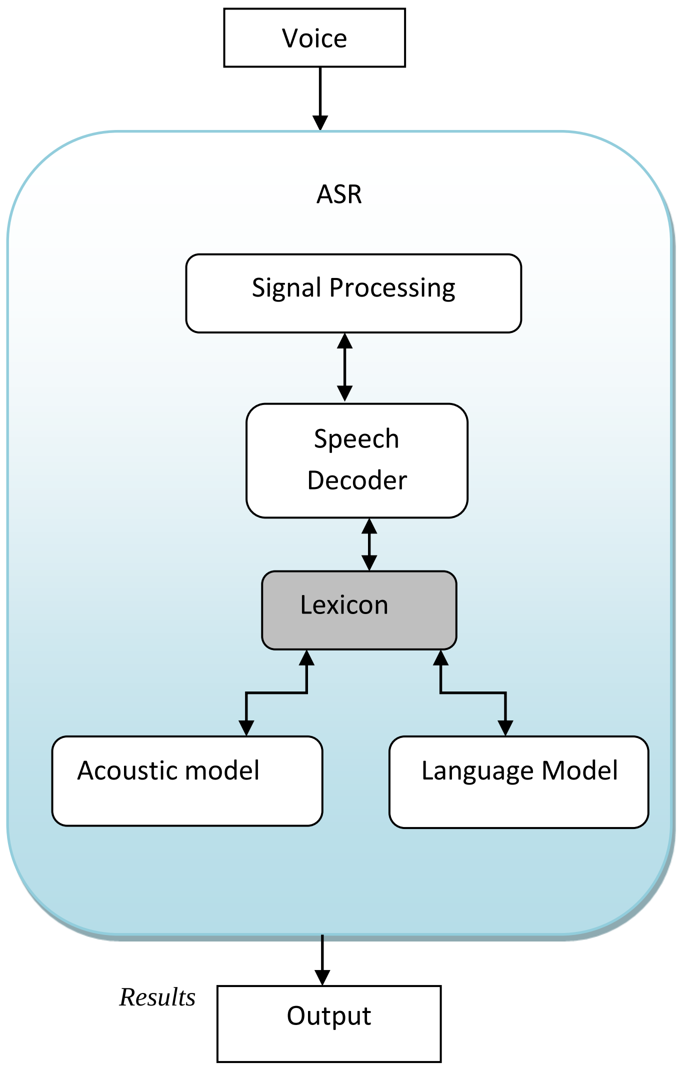
The ASR systems use the statistical acoustic and language models to find the most probable word sequence, $\hat{\mathbf{W}}$**,** that has a maximum posterior probability **P(W|X)** when using the Bayes theory.

$$\hat{W} = \arg_w \max \mathbf{P(W|X)}$$

$$= \arg_w \max \mathbf{P(X|W)P(W)\backslash P(X)} \ldots\ldots\ldots\ldots(1)$$

where **X** represent the acoustic feature sequence. **P(W)** is the language model.

Input Speech



Results

Figure 1: A Simple ASR System

In this research work, the hidden Markov models (HMMs) are given by P(**X**|**W**) - the acoustic model component. The Hidden Markov Model Toolkit (HTK) was used to build and manipulate the HMMs [5]. The HTK is mostly used for speech recognition but, it has also been used in many other pattern recognition applications that employ HMMs. The HTK has been developed by the speech recognition group at the Cambridge University (engineering department). It is designed for building the HMM- based speech processing systems, such as, an automatic speech recognition system.

The language model assigns a probability to a sequence of words by means of the probability distribution **P(W)**. It contains a set of rules for a language that is used as the primary context for recognizing words. The language model also, captures the regularities in the language. The complexity of the model is directly related to the size of the data on which it is trained.

During speech recognition, as depicted in Figure 1, the sequence of the symbols generated by the acoustic model is compared to a set of words in the lexicon, so as to produce the best sequence of the words that will generate the system's final output. It is at this stage that the rules that describe the restrictions present in the language should be introduced and this is achieved by making use of the language model in the system. The grammar contained in the language model can be made of the formal linguistic rules or the stochastic model. The formal linguistic rules can however bring up computational demands when included in the speech recognition system. This is what makes the stochastic models based on the probabilities more appealing for use in the speech recognition systems because of their simplicity [6].

The simplest statistical grammars are the *n-grams*. They predict the x_i word sequence based on the $x_{i-1}....x_{i-n}$ [7]. The n-grams are easy to apply, easy to interface with the speech recognition system and they are also the good predictors of the short term dependencies. In general, in an n-gram model, the probability of P(**X**) over a word string X = {x_1, $x_2 \ldots x_n$} that shows how frequently a string X occurs as a sentence is given by

$$\mathbf{P(X) = P(x_1, x_2, \ldots, x_n)}$$

$$\mathbf{= P(x_i)\, P(x_2|x_1)\, P(x_3|x_2, x_1) \ldots\ldots . P(x_n|x_1, x_2 \ldots x_{n-1})}$$

$$= \quad {}_{i=1}\prod^{n} \mathbf{P(x_i|x_1, x_2 \ldots x_{i-1})} \ldots\ldots\ldots\ldots\ldots\ldots\ldots(2)$$

where the $\mathbf{P(x_i|x_1, x_2 \ldots x_{i-1})}$ is the probability that x_i will follow from $\mathbf{(x_1, x_2 \ldots x_{i-1})}$

III. MORPHOLOGICAL APPROACH

Northern Sotho is an agglutinative tonal language. Nouns and verbs in the Northern Sotho language are created by means of the roots and affixes (prefixes and suffixes). Most of the roots are bound to morphemes because they cannot form words by themselves but require one or more affixes to complete the word [8]. In the morphological approach to the creation of a lexicon, the structure and the contents of the words were identified and analyzed. The Northern Sotho base verbs were analyzed. The morphological rules depending on the roots of the base verbs were developed. The morphological rules are developed using the scripting language Perl.

A. MORPHOLOGICAL RULES

Generally, in a verb, the most important part that expresses its meaning is the root. In the Northern Sotho language, the verb root and an inflectional ending are the ones that convey the unique meaning of a word [8].

Table 1 and Table 2 were used in this research to formulate base verbs and to yield a good and diverse root structure. Table 2 contains the vowels used in the Northern Sotho. Table 1 contains all the consonants in Northern Sotho. Each consonant was combined with every vowel in Table 2. For each combination, base verbs beginning with the syllable structure, CV (consonant-vowel), were formed [9]. For example, the first consonant in Table 1 is ’b’ and the following combinations of the CV syllable structure were formed: ‘*ba’*, ‘*be’* ‘*bi’*, *‘bo’, ‘bu’.* Some of the consonants in the table have two or three letters, for example, the consonant clusters ‘bj’ and ‘tlh’. Such consonants also, form the CV syllable structure when combined with each vowel. The reason for this is that, they in fact, represent complex/compound consonants in Northern Sotho [9]. The base verbs starting with these combinations were formed. Following this method and with the help from the lexicography unit in Northern Sotho at the University of Limpopo, 338 base verbs in Northern Sotho were formed.

b	j	psh	tsh
bj	k	pš	tšh
d	kg	pšh	tl
f	l	r	tlh
fs	m	s	w
fš	n	t	y
g	p	th	š
h	ph	ts	ng
hl	ps	tš	ny

Table 1: consonants used to form Northern Sotho base verbs

a	*e*	*i*	*o*	*u*

Table 2: Vowels in Northern Sotho

The basic meaning of a verb root can be modified in many different ways by adding suffixes and prefixes. In our approach, the verb roots that incorporate the extensions formed by adding the suffixes and the prefixes were used. Each morphological rule checks the root of the verb and modifies the meaning of the verb by adding the appropriate suffix/prefix in the context of the verb root to yield a new and valid word form. In this research project, ten extensions that form *ten new valid word forms* from a base verb were used. Having a hundred base verbs means this rule-based morphological approach can generate a thousand more inflected word forms. For example, using the verb ‘*rata’ – to love, ‘rat’* is the verb root:

extension: -w-

*rat + w → rat**w**a - be loved*

extension: -an-

*rat + an → rat**an**a – loving one another*

extension: -el-

*rat + el → rat**el**a - loving for something\someone*

extension: -eg-

*rat + eg → rat**eg**a – loved*

extension: -oll-

*rat + oll → rat**oll**a – undo love*

extension: - iš-

*rat + iš → rat**iš**a – cause to love*

extension: -ak-

*rat + ak → rat**ak**a – love repeatedly*

extension -išiš-

*rat + išiš → rat**išiš**a – love with passion\loving passionately*

extension: -i-

*i + rat → **i**thata – loving yourself*

extension: -n-

*n + rat → **n**thata – loving me*

However, not all verb roots take all the extensions because of the grammatical restrictions on the combinations of some of the roots and affixes. The morphological rules were able to come up with new and valid words that were not included in the existing Northern Sotho dictionary. The reason for the exclusion of some of these word forms may be attributed to their rare usage in the day-to-day spoken language.

IV THE PRONUNCIATION DICTIONARY

As depicted in Figure1, the ASR relies heavily on a lexicon. In this section, the process of creating an electronic Northern Sotho lexicon for the ASR systems is discussed.

Consonants and vowels in Table 1 and Table 2 respectively, were used as the phoneme set for the creation of the dictionary. These phonemes were clearly recorded using the *Praat* (freely downloadable program for speech analysis and synthesis) [10]. The phoneme set was validated. The base verbs together with the new words formed were used as the word list for the dictionary training. Table 3 below shows the grapheme set that was also used for the dictionary creation training. The symbol 'š' in Table 3 is represented by letter 'x' in the dictionary. The graphemes represent all the allowable letters in Northern Sotho.

a	g	l	r	y
b	h	m	s	š
d	i	n	t	
e	j	o	u	
f	k	p	w	

Table 3: Grapheme set used for the dictionary training

The dictionary was developed using the dictionary maker tool (DictMaker) that was designed at the University of Pretoria in collaboration with the Council for Scientific and Industrial Research (CSIR).

The wordlist, the phoneme set and the grapheme set were initialized to the Dictmaker without any pronunciation information. The DictMaker provides categories to sort the phoneme set for ease of use during the training. The DictMaker chooses a word from the wordlist and predicts its pronunciation. The audio version of the pronunciation is played back and the dictionary developer acts as a 'verifier'. The dictionary developer provides a verdict in context to the accuracy of the word-pronunciation pair. That is, the developer would decide whether the pair is correct, invalid, ambiguous or is unsure [11]. If the pronunciation is incorrect, the developer specifies the correct pronunciation by adding, removing or replacing the phonemes in the predicted pronunciation. Then the new audio would be played back again and the developer would verify if it is correct. When the word-pronunciation pair is declared correct, the system's learning algorithm extracts a new rule set and the rule set is added to the system. The rule set would be used to predict the next pronunciations. The more the rule sets in the system, the more the accurate predictions are provided. This process repeats until all the words in the wordlist are verified or adequate dictionary size is achieved.

When the dictionary creation training was done, each word-pronunciation pair was checked to ensure that the accuracy was not compromised. The Dictmaker uses the Default&Refine algorithm to extract the rule sets and to provide pronunciation prediction. In Default&Refine algorithm, for each grapheme, a default phoneme is derived as the phoneme to which the grapheme is likely to map. The words wherein the expected phoneme is not correct are now handled as refinements [12]. Table 4 shows the extracted rules that were frequently used during the dictionary training.

Rule	**Phone**	**Usage frequency**
-a-	a	1767
-k-	k	315
-l-	l	863
-m-	m	167
-n-	n	250
-o-	o	561
-p-	p	164
-r-	r	197
-t-	t	237
-u-	u	157
-w-	w	112
-x-	s_→	450

Table 4: Rule set in the dictionary

achieved after training with 2513 words

V. PRELIMINARY RESULTS AND DISCUSSION

DictMaker results statistics	
Number of words in dictionary	2505
Number of rules in the dictionary	92
Number of correctly pronounciated words	2504
Training words	2513
Accuracy	99.7%

Table 5: the overall statistics achieved after creating the pronunciation dictionary using the DictMaker Tool

An accurate dictionary was achieved through the use of the DictMaker tool. Table 5 shows the overall statistics obtained after the dictionary creation. Total of 2513 words were used for the creation of the pronunciation dictionary. 92 *Default&Refine* rule sets were extracted during the training. Out of 2513 words, 2504 were correctly pronounced in the dictionary. 99.7 percentage of accuracy was obtained by using the following formula:

Dictionary accuracy = 100% * (number of correctly pronounced words in the dictionary/number of the words training words) .

Hidden Markov Model Toolkit (HTK) was used to create a simple speech recognizer. The HMMs were continuous density mixture Gaussian tied state triphone with clustering performed using the phonetic trees [6]. The dictionary achieved from the dictionary training phase was used with this recognizer.

In total 130 sentences were recorded and used for the HTK speech recognizer training. Figure 2 depicts the results obtained. From Figure 2, out of 130 sentences, 39 (30%) were correctly recognized. Out of 1177 words, 752 (63.9%) were correctly recognized. There were three deletion errors (D), 255 substitution errors(S), and 214 insertion errors (I). Accuracy (Acc = 59.40) percentage is lesser than the correctness (Corr = 63.91) percentage because the Accuracy considers the insertion errors.

```
=========HTK Results Analysis ================

  Date: Thu May 05 15:00:40 2011

  Ref : testcount.mlf

  Rec : recout.mlf

------------------------ Overall Results --------------------------

SENT: %Correct=30.00 [H=39, S=90, N=130]

WORD: %Corr=63.91, Acc=59.40 [H=920, D=3, S=255, I=214, N=1177]
```

Figure 2: Results obtained from using the HTK-based speech recognizer system

The DictMaker tool provides a way of creating accurate pronunciation dictionaries. This tool can help in developing pronunciation dictionaries that are usable with automatic speech recognizers.

The results suggest that better accuracy can be achieved by improving the lexicon in the ASR system. The morphological approach to the development of the pronunciation dictionary has a significant impact in creating a larger and representative dictionary that can augment the existing dictionary in Northern Sotho.

VI. CONCLUSION AND FUTURE WORK

There is a need to develop the linguistic resources, especially for the under-resourced and the marginalized South African official languages. This is important in developing the country because the pronunciation dictionaries are a first and necessary requirement in designing the automatic speech recognition systems.

In the ultimate end, it is intended to create a pronunciation dictionary in Northern Sotho using the morphological rules on both the *verbs* and the *nouns* in the Northern Sotho language. Such a dictionary would improve the language coverage for the baseline ASR systems.

The best speech recognition accuracy would be achieved through the use of HMMs that provide an efficient way to build close parametric models. More training data with regard to building the statistical language model so that the speech recognition performance can be improved will be used.

VII. ACKNOWLEDGEMENT

We acknowledge the contribution of the Northern Sotho National Lexicography Unit at the University of Limpopo (Turfloop Campus) for their kind help with the formation of Northern Sotho verbs and the validation of the phoneme set for the training of the pronunciation dictionary. We also acknowledge the University of Limpopo Telkom Centre of Excellence for Speech Technology for funding this research project.

VIII. REFERENCES

[1] E.S Bosch, J. Jones, L. Pretorius, W. Anderson, "*Resource Development for South African Bantu Languages: Computational morphology Analyzer and Machine readable Lexicons*", In the Proceedings of the Workshop on Networking the Development of language resources for African language 5Th International Conference on Language Resource and Evaluation, Genoa, Italy, May 2006, pp.38-43.

[2] Biadsy Fadi, Nizar Habash, Julia Hirschberg, "*Improving the Arabic Pronunciation dictionary for Phone and word Recognition with Linguistically-Based Pronunciation rules*", In the Proceedings of the North American Association for Computational Linguistics, Boulder, Colorado, USA, 2009, pp.397-405.

[3] Sepedi Language: http://www.kwintessential.co.uk/language/about/sepedi.html, accessed: 04 May 2011

[4] L.R Rabiner, B.H Juang, "Fundamentals of speech recognition", Prentice Hall, Englewood cliffs, N.J, 1993.

[5] S. Young, D. Kersaw, J. Odell, D Ollason, V. Valtchev, P.C Woodland. The HTK book, (for version 3.4.1) *Machine Intelligence Laboratory*,

Department of Engineering University of Cambridge, U.K. 2002

[6] M. Meeter, J. R Rohlick, “Statistical Language Modelling Combining N-gram and Context Free Grammars”, In Proc. of International Conference on Acoustic, Speech and Signal Processing., Vol. II, pp.37-40, 1993.

[7] Huang H., Acero A., Hon H., Spoken Language Processing, Prentice Hall, Upper Suddle River, N.J., 2001.

[8] G. Poulos, J. Lawrence. A Linguistic Analysis of Northern Sotho, GoodWood, 1994, pp.115-244

[9] C.S Price, Q.H. Gee. (1988). Syllable Analysis of North Sotho and its Effect on Computerized Hyphenation. South African Journal of Science, Vol. 84, pp.480-482.

[10] Praat program: http://www.fon.hum.uva.nl/praat/ accessed: 8 Feb 2011

[11] M. Davel, M. Peche. (2006, Sept). Dictionary maker User Manual, Meraka Institute. Pretoria, South Africa. [Online]. Available: http://dictionarymaker.sourceforge.net/

[12] M. Davel and E. Barnard, “*A default-and-refinement approach to pronunciation prediction,”* in Proceedings of the symposium of the Pattern Recognition Association of South Africa, South Africa, Nov 2004, pp.119-123.

VIII. BIOGRAPHY

Mpho Nkosi holds an honors degree in Computer Sciences from the University of Limpopo and is currently studying towards an MSc degree in Computer Sciences at the same institution. She’s attached to Telkom Centre of Excellence for Speech Technology. Her research interests are in Automatic Speech Recognition, Computer Networks and Computer Security.

Towards a comprehensive pronunciation dictionary for automatic Speech recognition using morphological Approach

MC Nkosi[1], MJD Manamela[2] and N Gasela[3]

University of Limpopo, Department of Computer Science, Private Bag X1106, Sovenga, 0727

e- mail:mphocaselina@ymail.com

Abstract

The pronunciation dictionaries or lexicons play an important role in guiding the predictive powers of the Automatic Speech Recognition (ASR) systems. As the use of automatic speech recognition systems increases, there is a need for the development of dictionaries that cover a large number of inflected word forms to enhance the performance of ASR systems. This paper describes the morphological-driven approach to the creation of a more comprehensive and broadly representative Northern Sotho (NS) pronunciation dictionary for ASR systems to augment the existing data-driven methods of dictionary creation.

Introduction

ASR is a process through which a computer system can recognize the words spoken by a person and coverts them to text. The ASR technology has recently gained popularity in the modern information society, especially in the customer services companies. A well-trained basic ASR system can recognize single word entries such as yes/no responses and spoken words. As the use of the ASR systems in spoken language processing is increasing, there is a great need for the development of lexicons that cover a large number of inflected words for the under-resourced official languages of South Africa. The following section discusses the methodology used in developing the lexicon and a simple speech recognizer for Northern Sotho. Preliminary results achieved are discussed in the last section.

Methodology

Northern Sotho is an agglutinative tonal language. Each verb can systematically and rapidly generate more valid inflected word forms in Northern Sotho through the concatenation of the root of a verb with appropriate affixes. The base verbs together with morphological rules were used to generate more valid inflected word forms. Phoneme set was selected and recorded as audio recordings using *Praat (program for speech analysis and synthesis)*. For each verb, morphological rules were applied in order to match certain conditions on context of the verb and then more valid inflected words were generated through the concatenation of relevant affixes. The base verbs together with the new Northern Sotho Words formed were used for training the pronunciation dictionary using the Dictionarymaker (Dictmaker) tool [1]. The created Dictionary was incorporated within a simple speech recognizer that was developed using the Hidden Markov Model toolkit (HTK) freely downloadable from engineering department of the University of Cambridge[2].

Preliminary Results and Discussion

```
___________HTK Results Analysis____________________________

Date: Wed Aug1 10 11:32:41 2011

Ref: testref.mlf.mlf

Rec: recount.mlf

_____________Overall Results________________________________

SENT: %Correct=44.00 [H=88, S=112, N=200]

WORD: %Corr=82.07, Acc=63.15 [H=755, D=10, S=155, I=174, N=920]

____________________________________________________________
```

Figure1:results obtained from using HTK-based speech recognizer

DictMaker results statistics	
Number of words in dictionary	2505
Number of rules in the dictionary	92
Number of correctly pronounciated words	2504
Training words	2513
Accuracy	99.7%

Table 1: The overall statistics achieved

An accurate pronunciation dictionary was achieved through the use of the Dictmaker tool. Table 1 above shows the overall statistics achieved after creating the pronunciation dictionary using the Dictmaker tool. A total of 2513 words were used for the creation of the pronunciation dictionary. 92 *Default&Refine* rule sets were extracted during the training. Out of 2513 words, 2504 were correctly pronounced in the dictionary. This gave 99.7 percentage of accuracy. For the HTK speech recognizer training, we recorded and used 200 sentences. Figure 1 depicts the results obtained. Out of 200 sentences, 44% were correctly recognized. Out of 920 words, just over 82% were correctly recognized. Accuracy (Acc = 63.15) percentage is lesser than the correctness (Corr = 82.07) percentage because the Accuracy considers the insertion, substitution and deletion errors. These results suggest that better and higher recognition accuracy can be achieved by improving the lexicon in the ASR system. The morphological approach to the development of

[2] http://htk.eng.cam.ac.uk

the pronunciation dictionary has a significant impact in creating a larger and representative dictionary that can augment the existing Northern Sotho pronunciation dictionaries.

References

[1] M. Davel, M. Peche. (2006, Sept). Dictionary maker User Manual, Meraka Institute. Pretoria, South Africa. [Online]. Available: http://dictionarymaker.sourceforge.net/