University of Fort Hare

Together in Excellence

Development of a Stemmer for the IsiXhosa Language

by

Mnoneleli Nogwina

A dissertation submitted in fulfillment of the requirements for the Degree

Master of Science

in the

Department of Computer Science

University of Fort Hare

Supervisor: Mr. Zelalem. S. Shibeshi

Co-Supervisor: Dr. Zoliswa. O. Mali

Submitted: January 2015

Declaration

I, Mnoneleli Nogwina (Student Number: 200904717), the undersigned, hereby declare that the work contained in this dissertation is my own original work and has not previously been submitted at any educational institution for a similar or any other degree. Information extracted from other sources is acknowledged accordingly.

Signature……..

Date……..

Acknowledgements

I would like to thank God Almighty who has made it possible for me to complete this study and for being the shining light over my life.

I would like to thank Mr. Zelalem Shibeshi and Dr. Zoliswa Mali for their everlasting support. Without you next to me it would not been possible to complete this two years of Masters. Thank you once again a million times for the advice, meetings and your constructive criticism.

I would also like to thank the Telkom Centre of Excellence and NRF for the financial support throughout my education at the University of Fort Hare.

To my family: Miss Noluthando Soga, Nokonwaba Soga, Mduduzi Nogwina and Anelisa Nogwina, thank you very much for your understanding and support throughout my education. Your love is what kept me going, may the Lord Almighty watch over you and keep that spirit of togetherness burning for as long as we may live.

Finally I would like to thank everyone who has supported me during this study. Thank you all.

Dedication

This work is dedicated to my mother Noluthando Soga. She has been the source of inspiration for all my life. I know it wasn't easy raising three children on your own but you are the strongest single parent I know. Thank you *Mazulu*, I love you. This work is also dedicated to my daughter Muhle Ntshobane and my siblings Mduduzi Nogwina and Anelisa Nogwina. I know it was not easy when you guys had to spend holidays without me. Without all of you I would not be where I am today.

Publications

Nogwina, M; Shibeshi, Z; Mali, Z, & Moroosi, N. (2013). *Development of a Stemmer for the isiXhosa language*. WIP. SATNAC Conference 2013, 01-04 September 2013, Spier Wine Estate, Stellenbosch, Western Cape, South Africa.

Nogwina, M; Shibeshi Z; & Mali, Z. (2014). Towards *Developing a Stemmer for the isiXhosa language*. SATNAC Conference 2014, 31 August -03 September 2014, Boardwalk, Port Elizabeth, Eastern Cape, South Africa.

Acronyms and Abbreviations

AI	Artificial Intelligence
CI	Conflation Index
DI	Distinctness Index
FOSS	Free and Open Source Software
HMM	Hidden Markov Models
IPA	International Phonetic Alphabet
IR	Information Retrieval
NLP	Natural Language Processing
OI	Over-Stemming Index
UI	Under-Stemming Index
YASS	Yet Another Suffix Stripper

Scope of the Dissertation

The tittle of the dissertation says “Development of A Stemmer for the isiXhosa Language”.

The dissertation present the noun Stemmer, meaning the scope of the research only focuses on the nouns of the isiXhosa language. The objectives clearly states that we studied the noun formation in isiXhosa, how the prefixes and suffixes are attached to root words to form nouns. After we learnt the noun formation we also focused on how the stems are formed from removing affixes and applied this theory in the development of the Stemmer.

Abstract

IsiXhosa language is one of the eleven official languages and the second most widely spoken language in South Africa. However interms of computational linguistics, the language did not get attention and natural language related work is almost non-existent. Document retrieval using unstructured queries requires some kind of language processing, and an efficient retrieval of documents can be achieved if we use a technique called stemming. The area that inlvolves document storage and retrieval is called Information Retrieval (IR). Basically, IR systems make use of a Stemmer to index document representations and also terms in users' queries to retrieve matching documents. In this dissertation, we present the developed Stemmer that can be used in both conditions. The Stemmer is used in IR systems, like Google to retrieve documents written in isiXhosa. In the Eastern Cape Province of South Africa many public schools take isiXhosa as a subject and also a number of Universities in South Africa teach isiXhosa. Therefore, for a language important such as this, it is important to make valuable information that is available online accessible to users through the use of IR systems. In our efforts to develop a Stemmer for the isiXhosa language, an investigation on how others have developed Stemmers for other languages was carried out. From the investigation we came to realize that the Porter stemming algorithm in particular was the main algorithm that many of other Stemmers make use of as a reference. We found that Porter's algorithm could not be used in its totality in the development of the isiXhosa Stemmer because of the morphological complexity of the language. We developed an affix removal that is embedded with rules that determine which order should be followed in stripping the affixes. The rule is that, the word under consideration is checked against the exceptions, if it's not in the exceptions list then the stripping continue in the following order; Prefix removal, Suffix removal and finally save the result as stem. The Stemmer was successfully developed and was tested and evaluated in a sample data that was randomly collected from the isiXhosa text books and isiXhosa dictionary. From the results obtained we concluded that the Stemmer can be used in IR systems as it showed 91 % accuracy. The errors were 9 % and therefore these results are within the accepted range and therefore the Stemmer can be used to help in retrieval of isiXhosa documents. This is only a noun Stemmer and in the future it can be extended to also stem verbs as well. The Stemmer can also be used in the development of spell-checkers of isiXhosa.

Keywords: IsiXhosa, Stemming, Stemmer, Stemming Algorithm and Information Retrieval

Table of Contents

Declaration i
Acknowledgements ii
Dedication iii
Publications iv
Acronyms and Abbreviations v
Scope of the Dissertation vi
Abstract vii
Table of Contents viii
Table of Figures xii
List of Tables xiii
List of Listings xiv
1. Introduction and Background 1
1.1 Introduction 1
1.2 Background on Natural Language Processing 1
1.2.1 Data organization and retrieval 1
1.2.2 Information storage and retrieval 2
1.3 Problem Statement 3
1.4 Research Questions 4
1.5 Research Objectives 4
1.6 Research Methodology 4
1.7 Motivation 6
1.8 Outline of Dissertation 6
1.9 Conclusion 7
2. Literature Review 9
2.1 Natural Language Processing 9
2.1.1 Morphology 9
2.1.2 Syntax 9
2.1.3 Semantics 9
2.1.4 Pragmatics 10
2.2 Ambiguity in NLP 10
2.3 Applications of NLP 10

2.4 Conflation Techniques 11
2.4.1 Affix removal method 12
2.4.2 Successor variety method 12
2.4.3 Table lookup method 13
2.4.4 N-gram method 13
2.5 Stemming algorithms 14
2.5.1 Rule-Based Approach 14
2.5.2 Statistical/Machine Learning Approach 15
2.6 Related Work 16
2.6.1 Stemming Algorithms for English Language 16
2.6.2 Stemming Algorithms for Other Languages 23
2.7 Evaluation Methods for Stemming Algorithms 26
2.8 Conclusion 27
3. IsiXhosa Language 29
3.1 Introduction 29
3.2 History of the isiXhosa Language 29
3.3 IsiXhosa Orthography 30
3.3.1 International Phonetic Alphabet 31
3.4 IsiXhosa Noun Classes 33
3.4.1 IsiXhosa Noun Class Prefixes 33
3.4 The Role of the Prefix in IsiXhosa Parts of Speech 38
3.5 The Role of Prefix in Noun formation 38
3.5.1 Vocative 39
3.5.2 Locatives 39
3.5.3 Copulatives/ Predicative 39
3.5.4 Negative Copulatives/ Predicative 40
3.6 IsiXhosa Roots 40
3.7 IsiXhosa Suffixes 41
3.8 Conclusion 42
4. System Design 44
4.1 Introduction 44
4.2 Use Case Diagrams 44

4.2.1 User Roles 44
4.2.2 Use Cases for System Administrator 44
4.2.3 Use Cases for General Users 45
4.3 The isiXhosa Stemming Algorithm 46
4.3.1 System Flow Diagram 46
4.3.2 System Back-end Design 48
4.3.3 The Rules 49
4.4 Conclusion 53
5. Implementation 55
5.1 Introduction 55
5.2 Implementation 55
5.3 Modules of the Stemmer 56
5.3.1 Exceptions List Checking 56
5.3.2 Prefix Stripping 56
5.3.3 Suffix Stripping 57
5.4 Putting All Together 58
5.5 Conclusion 59
6. Testing, Results and Discussion 61
6.1 Introduction 61
6.2 System Testing 61
6.2.1 Preprocessing 61
6.2.2 Test Data 61
6.2.3 Functionality Testing 63
6.3 Evaluation 63
6.4 Results 64
6.5 Discussion 65
6.6 Problems Related To Noun Classes 66
6.6 Conclussion 68
7. Conclusion 70
7.1 Introduction 70
7.2 Research Summary 70
7.3 Achievements of the Objectives 71

7.4 Future Work 72
8. References 74
9. Appendix A-Results 78
10. Appendix B- Test Data 91

Table of Figures

Figure 2-1: Conflation Techniques. *12*
Figure 2.2: Lovins Stemming Algorithm (Lovins, 1968). *18*
Figure 2.3: Paice/ Husk Stemming Algorithm (Paice & Husk, 1987). *21*
Figure 2.4: Porter's Stemming Algorithm (Porter, 1980) *22*
Figure 2.5: Automated Morphological Processor (AMP) Overview (Fotinea et al., 2001). *24*
Figure 3.1: International Phonetic Alphabet. *32*
Figure 4.1: Administrator Use Case Diagram *44*
Figure 4.2: General User Use Case Diagram *45*
Figure 4.1: Front-end and Back-end architecture *48*
Figure 5.1: StemXho: The IsiXhosa Stemmer *55*
Figure 6.1: Test Results *64*
Figure 6.2: Distribution of Errors *65*

List of Tables

Table 3.1: Forms of Nouns. *39*

Table 3.2: Word formation in isiXhosa. *41*

Table 5.1: Prefix list. *57*

Table 5.2: Suffix list. *57*

Table 6.1: Sample Test Data. *62*

Table 6.2: Sample Test Data Showing the Feminine, Diminutive and Augmentative. *63*

Table 6.3: Results *64*

List of Listings

Listing 5.1: Reading a File *58*

Listing 5.2: Prefix and Suffix Removal for the Second (2) Noun Class *58*

Chapter One

Introduction and Background

1. Introduction and Background

1.1 Introduction

We introduce the area of research we carried out including the problems that led to the introduction of this research project in this Chapter. We also present background information together with the objectives of the research and methodology, accordingly. Section 1.2, discuses natural language processing, how stemming fits into this category is reviewed, data organization and retrieval is discussed, information storage and information retrieval is presented. In Section 1.3, the problem statement is presented. In Section 1.4 research questions are presented. In Section 1.5, the research objectives are presented. In Section 1.6, the methodology followed when conducting this research is outlined. In Section 1.7, the research motivation is outlined. The Chapter also presented the dissertation outline in Section 1.8 and Conclusion in Section 1.9.

1.2 Background on Natural Language Processing

Natural languages are used in written and spoken form by humans to communicate and transfer information among them. Currently we are not yet at that point where these can be understood by computers in their unprocessed forms. There is a field of study that tries to achieve this and is called Natural Language Processing (NLP). NLP can be defined as the collection of techniques employed to try and accomplish this goal of computers understanding of the spoken and written languages of human in their unprocessed forms(Ahmad, 2007). However, achieving this is not an easy task, because natural languages have much ambiguity, meaning one word can have several different meanings in different contexts (Tomita, 1985). In this project the problem we have in processing queries submitted in natural language to retrieve relevant documents from a collection of documents from an information retrieval system and the solution to this problem is discussed. Stemming is one technique of NLP that we use to solve this type of a problem, and a Stemmer for isiXhosa language is developed and presented.

1.2.1 Data organization and retrieval

One of the most popular ways of organizing data is keeping it in a database system. In a database system, data is stored in an organized manner which makes retrieval easy. However, one can only search for data or information that is only stored in a database and as is. The database management system also uses a particular syntax for data retrieval and if one uses a wrong

syntax they may get a wrong output. Data is stored in tables and arranged using indexes. Especially fields that are going to be used for searching are indexed so that searching for a particular term or value is easy. These tables in a database are also related to one another and therefore making it easy to go from one table to another because of the relationships. This is different from an online information retrieval system where documents are written in natural language with all the different variation of terms to explain a particular term or concept. The queries are also submitted using natural language, again with all the variations, and therefore finding matching term may be difficult. In order to get the required documents, terms representing documents and also queries needs to come to common form so that searching is easier. The following section discusses how information storage and retrieval works.

1.2.2 Information storage and retrieval

When one searches for a document online he/she submit a query by assuming a matching word(s) that can be found in the document he/she is looking for. After the query is submitted, a corpus of documents is checked for a matching document that contains the matching word(s) and those documents that meet these criteria are returned as output. There are different ways of doing this; one of them is checking a document word by word but as mentioned above words could exist in different forms and a word used in a query may not exist in a document that the user is looking for. So we need to introduce a technique that brings the different variations of words into common form. In other words, words need to be changed into their root form (stem). Stemming is employed to achieve this because in stemming documents are indexed using stems of the keywords and all the documents that have same stems are indexed together. So when a query comes, all the matching documents are retrieved because retrieval systems also change the words used in queries into their stem before doing the matching. Consider for example, a document that contains information related to fish. The words that the document may contain could be any of the words that relate to fish, like fishes, fished, fish, fisheries, and fishing. So if we use one of these words in our query but a given document do not contain that particular word, it may not be retrieved but if we use stemming to index documents then all the above words could be stemmed to fish and if a user asks for fishes all these documents that contain the stem word fish are retrieved as they fall under one stem (fish). This process helps in query broadening.

So Information Retrieval (IR) systems use a tool called Stemmer to index documents using stem words and also when processing users' queries they change the users' queries into stem words and do the matching. Therefore, a Stemmer is an important tool for any IR system and a Stemmer is developed to handle the indexing of isiXhosa documents so that IR systems that contain documents written with isiXhosa will be effective in document retrieval activities.

1.3 Problem Statement

We are living in the information age and it is important for everyone to have information in their fingertips in any language and especially languages of their own. Currently, most information is available on the Internet and can be accessed through search engines online. In the Eastern Cape region of South Africa isiXhosa is taken as a subject in public primary and secondary schools. IsiXhosa is also the medium of instructions in these schools. The language is taught as one subject from elementary schools up to matric level. There is a lot of learning material available for the language and with the advancement of technology we see the online classes and other learning applications being developed. There are also several documents, like stories, news items, and novels written in isiXhosa. For a language important as isiXhosa it is obvious to see the need for information retrieval system. Such an IR system may help learners to search for information online and to our knowledge so far there is no IR system that is localized for the isiXhosa language.

The isiXhosa language, like any other language contains words with different forms and the searching of documents will be very difficult. As mentioned above, IR systems make use of the Stemmers to index the documents available in the corpus and there is no Stemmer that was developed for the isiXhosa language. Because of this we saw the need to investigate and develop a Stemmer for the isiXhosa language. In other words there is a need for a Stemmer that will help organize the documents in the corpus to make sure that the information returned to users' queries is as broad and accurate as possible. This Stemmer can be plugged into an IR system that has isiXhosa documents to help with the query broadening and enhance the work of governmental institutions like Education Department of the Eastern Cape. In general, the following are our research questions.

1.4 Research Questions

Can the existing stemming algorithms be used in the development of a Stemmer for isiXhosa, bearing in mind that stemming algorithms are language dependent?

What are the important aspects to consider in developing a stemming algorithm for the isiXhosa language?

How to design and implement stemming algorithm?

How to evaluate the proposed system?

1.5 Research Objectives

The following are the main objectives for undertaking this research project:

- To dertemine important aspects to consider in algorithm development.
- To design and implement stemming algorithm.
- Investigate the rules of isiXhosa as a written language and the noun formation in the language.
- Finding a suitable stemming algorithm that can be used as a reference in building the isiXhosa noun Stemmer.
- Develop a noun Stemmer for isiXhosa.
- To evaluate the proposed system.

1.6 Research Methodology

Developing a Stemmer will require extensive reading and visiting a lot of literature review, gathering information from other Stemmers and stemming algorithms for other languages. In general we followed the following methods to develop the Stemmer:

- **Literature review**

Studying the language's morphology is the most important component of this research and therefore a literature survey was used to collect information and to understand the language. Understanding the language includes understanding the word formation of the language, how the prefixes and suffixes are attached to root words and how they are removed to form stems, and also identifying exceptional cases. A literature survey of the already existing Stemmers for other languages has been made and used as a reference in developing the Stemmer we are presenting

in this dissertation. In general, the main purpose of literature review was to equip the researcher with the background information of this area of study and to furnish the researcher with the current information on the research that has been undertaken in the area of IR and stemming. The output of this phase was presented in detail in Chapter Two and Chapter Three. The literature review was gatherd from online articles, websites, text books and the isiXhosa dictionary

- **Algorithm Development**

The algorithm development is done iteratively to include all the rules and to open room for improvements. The development is started by evaluation of existing stemming algorithms.

- ✓ **Evaluation of Existing Algorithms**

A literature survey on the existing algorithms was conducted. We studied the Lovins stemming algorithm, Porter stemming Algorithm, Paice/Husk stemming algorithm, Dawson stemming algorithm and the Greek stemming algorithm. This was done so as to choose the most appropriate algorithm that can be applied in the development of the isiXhosa stemming algorithm.

- ✓ **Developing an Algorithm**

All the rules of the algorithm were applied; different tables, including exception list checking, prefix removal and suffix removal were set up in the development of the algorithm. The algorithm flowchat was followed so as to not miss the rules as stated by the researcher.

- **Development Tools**

`The Stemmer for the isiXhosa was developed on Linux Operating System using Netbeans IDE and Java programming language. The reason for choosing Java is that string manipulation is easy in Java and the researcher is also familiar with Java.

- **Evaluation, this part checks if the Stemmer stems the nouns and check if the acceptable standards are met by the developed Stemmer**.

To evaluate the performance of the Stemmer we considered manual method. In the manual method, a human expert/linguist is called to decide the correct stem for every word. A linguist in

this method checks the number of correct results, and the errors occured due to over-stemming and under-stemming. A linguist was given both files; the one with nouns before they were stemmed and the one with stems. The linguist had to apply her expertise in deciding which stems were correct and which ones which were not. The details of this section are discussed in Chapter Six.

1.7 Motivation

The number of Internet users is growing even in the rural areas where most people cannot read or write English and yet they have access to the Internet. Therefore, this research project is intended for users who might be interested in finding information on the Internet using their mother tongue (IsiXhosa). In fact, the medium of instruction for elementary schools in Eastern Cape is isiXhosa and there are lots of materials written using this language at various times. As mentioned before isiXhosa is also one of the 11 official languages in South Africa and we expect that the document collection written in isiXhosa on the Internet will increase through time. There has been no study of this nature before for the isiXhosa language and for me as a researcher it is interesting to conduct the first study that is going to offer so much for the isiXhosa speaking people.

1.8 Outline of Dissertation

Chapter Two – Literature Review, this Chapter focuses on the related works. It explains the existing conflation techniques, as well as English stemming algorithms and concludes by discussing the non-English stemming algorithms.

Chapter Three – The isiXhosa Language, this Chapter explains the isiXhosa language and its morphology. We first look at the history of the language, then word formation and how these words are formed by attaching different prefixes and suffixes to the root/stem of a word, specifically nouns.

Chapter Four – System Design, this Chapter explains the design of the system. The architecture of the system its internal structures and the rules are explained in this Chapter.

Chapter Five – Implementation and Testing, this Chapter begins by presenting the steps taken in the implementation of the Stemmer that we are reporting here and further explains the

methods that were followed when testing and evaluating the system. The Chapter describes how the testbed was set up and how the results were obtained.

Chapter Six- Evaluation, Results and Discussion, this Chapter focuses on the evaluation of the system and the results that were obtained. It discusses the results and compare with the objectives.

Chapter Seven- Conclusion, this Chapter presents the research summary and as well as the achievement of the objectives. The Chapter concludes the research and talks about the future works that can be done in the future.

1.9 Conclusion

This Chapter presented the introduction of this research project as a whole. It has given the background and further explains what is stemming. Data organization and data retrieval is also discussed in detail in this Chapter. Another focus area of the Chapter was the problem statement. A list of objectives to be achieved and the methodologies to be followed in order to achieve these objectives are also outlined. The following Chapter provides a detailed literature review of this study.

Chapter Two

Literature Review

2. Literature Review

2.1 Natural Language Processing

As defined in Cherpas (1992) Natural Language Processing (NLP) is "a part of Artificial Intelligence (AI) concerned with endowing computers with verbal and listener repertoires, so that people can interact with them more easily". This field is also known as computational linguistics. Ahmad (2007) mentioned that there are many advantages of using natural language as a communication channel between man and machine, one of them is the fact that humans already know natural language and therefore they do not need to learn an artificial language in order to communicate with the machine. NLP involves text and voice processing, but the main focus of this study is in text processing. NLP consists of four main components of text processing, which are: Morphology, Syntax, Semantics and Pragmatics.

2.1.1 Morphology

Morphology is a component of NLP that deals with smallest units of meaning or componential nature of words which are composed of morphemes (Liddy, 2001). Morphology plays a very important role in identifying parts of speech in a sentence and words that interact together. It checks how the broken components of the words affect their grammatical status. For example, the word Preregistration – can be broken down into the prefix "Pre-", the stem or root – "registra-" and the suffix – "-tion".

2.1.2 Syntax

This component takes the language of interest or the targeted language's grammar and applying its rules and determines the role of each word in a sentence (Ahmad, 2007). Its main focus is on analyzing the words in a sentence to reveal the grammatical structure of a sentence. For example, in English we may have "The boy eats his food", this is a standard **subject-verb-object**. If one switches the order, the meaning of this sentence may completely change.

2.1.3 Semantics

This is the component that determines the possible meanings of a sentence by focusing on the interactions among word-level meanings in the sentence (Liddy, 2001). It is also regarded as a conveyor of useful information relevant to the scenario as a whole.

2.1.4 Pragmatics

This component is more concerned with overall purpose of the statement being analyzed. Pragmatics is defined as "the analysis of the real meaning of an utterance in human language, by disambiguating and contextualizing the utterance". This goal is accomplished by identifying ambiguities encountered by the system and resolving them using one or more types of disambiguation techniques (Ahmad, 2007). The following section discusses the different types of ambiguities that may occur during language processing.

2.2 Ambiguity in NLP

Ambiguity is defined as "the problem where one word is having more than one meaning in a language in which the word belongs" (Geeraerts, 2006). The issue of ambiguity is the major problem in most natural language processing systems. It tends to bring difficulties during the developmental stages of a system because it has to be taken into consideration and therefore systems designed accordingly and with the exceptions of such cases of ambiguity. As mentioned in Geeraerts (2006) there may be different types of ambiguities; such as Lexical ambiguity, where one word has more than one possible meaning; Syntactic ambiguity, where more than one parse of a sentence exists; Semantic ambiguity, where more than one possible meaning of a sentence exists; Referential ambiguity, where something is referred to without explicitly naming it using pronouns such as "he", "it" and "they" and finally Local ambiguity, where one part of the sentence is unclear but resolved when the whole sentence is examined. In this research, we encounter Lexical ambiguity because we are dealing with isiXhosa nouns where each noun may have different meanings in different situations where a noun is used. In the following section we discuss the applications of NLP.

2.3 Applications of NLP

Natural language processing can be applied in many areas and some of the proven abilities of NLP include the ability to handle simple pronouns, answer direct questions, correct spelling errors and coordinate multiple files. NLP is also applied to duplicate error report detection, tutoring systems, speech recognition and synthesis, database interfaces and information retrieval systems. Several people utilized NLP techniques to develop various systems. For example, Alexandersson (2007) developed a prototype that was used to identify duplicate error report. In the prototype, the author identified 40% of the total number of duplicate error reports submitted

to the users of this prototype. The prototype used Tokenization, Stemming, Stop Word Removal, Vector Space Calculation and Similarity Calculation to identify the duplicates. In one of his tutorials Hendrix (1981) suggested that accessing databases is one of the most important areas for the practical application of NLP. The next Section discusses the conflation techniques.

2.4 Conflation Techniques

Conflation is defined as the process used by search engines to treat words with the same stem as synonyms as a kind of query broadening (Ali and Ibrahim, 2012). Ali and Ibrahim (2012) also mentioned that the effectiveness of searching would be expected to increase if it was possible to conflate the variants of a word to a single variant. As explained by Willett (2006), natural language texts contain many different variants of a basic word (e.g., Computer, Computational, Computers, and Computing). Conflation of words can be done manually by checking a document word by word or automatically by using Stemmers (Sharma, 2012). Documents are related to each other by the terms they have in common and the more terms they have in common the more they are related. In the English language and many related languages, morphological variation of words takes place at the right-hand end of the word form. This is different in the isiXhosa language. IsiXhosa has a rich set of prefixes which are found at the beginning of the word. Of course IsiXhosa also has suffixes that are found at the right-hand end of a word form.

As mentioned in Smirnov (2008) the ability of an information retrieval system to conflate words allows the reduction of index and helps improve recall sometimes without reducing the precision of the system. Recall and precission are IR terms used to test the performance and correctness of a Stemmer. Sharma (2012) also justifies that conflating word forms reduces the size of index files, since one stem corresponds to many full terms and by storing stems instead of full terms compression factor of 50% can be achieved. Basically, there are four automatic approaches of conflation, namely Affix removal, Successor variety, Table lookup and N-gram methods. These methods are represented diagrammatically in Figure 2.1.

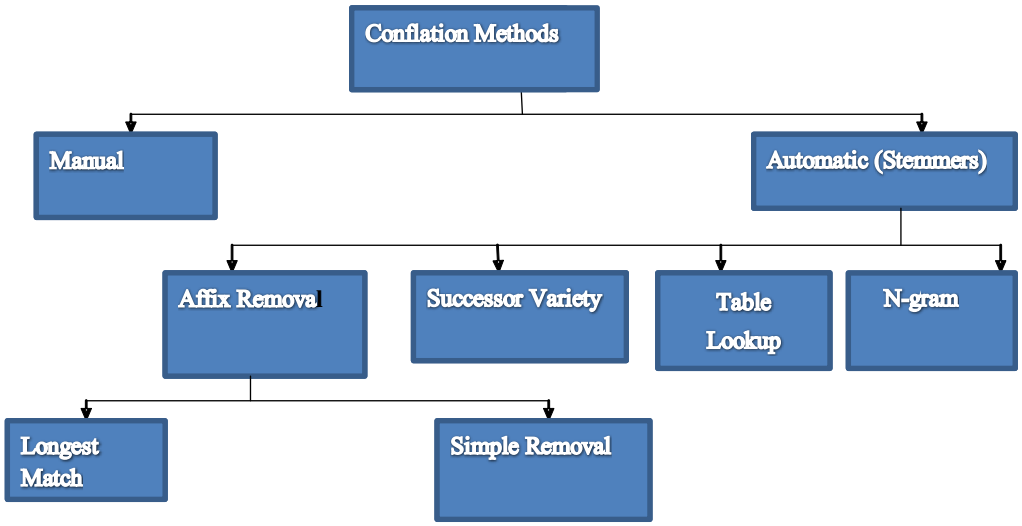

Figure 2-1: Conflation Techniques.

2.4.1 Affix removal method

Affix removal method removes suffixes and prefixes from a word so as to retain the stem (Sharma, 2012). This method is based on two principles longest match and simple removal. This method is the most widely used method for most Stemmers. Simple removal is an iterative stemming algorithm that removes strings in each order class one at a time, starting at the end of the word working towards the beginning. When using affix removal methods, there is a concept called order class which refers to the arrangement of affixes to stems of words (Sharma, 2012). Smirnov (2008) shows that no more than one match is allowed within a single order class and his argument is based on the fact that suffixes are attached to stems in a certain order. Lovins (1968) stated that the longest match technique suggest that within any given class of endings, if one ending provides a match, then the one which is longest should be removed. Porter (1980) also used this approach in developing a Stemmer for the English language.

2.4.2 Successor variety method

This technique uses the frequencies of letter sequence in a body of text as the basis of stemming (Abode, 2012). This can be further defined as the number of different characters of a string that

follow it in some body of text. Suppose we have the following body of text: **back, beach, body, backward, boy**. If one had to find the successor variety of **"battle"** the procedure would be to take the first letter **"b"** and check from the body of text how many times **"b"** is followed by **"a", "e" and "o"** and the other **"a"** and **"o"** from **backward** and **boy** are left out because they are repeating and therefore successor variety of battle is three (Sharma, 2012). In general this method has three parts; the first part is to determine the successor variety of a term, the second part is to segment the term using the frequencies of letter sequence and the last part selects one of the segments as the stem (Al-Shalabi *et al.*, 2005).

2.4.3 Table lookup method

This method uses a table to store all indexed terms and their corresponding stems so that terms from queries and indexes could be stemmed very fast. Then stemming is performed via lookups in the table (Sharma, 2012). This method poses some problems as the table may take too much storage space and it requires a lot of work in the language and these tables may also miss out some exceptional cases (Abode, 2012).

2.4.4 N-gram method

This is another statistical method developed by Adamson & Boreham (1974). In this approach pairs of words are associated on the basis of "unique digrams" that they both possess. "N" can be 2, 3, etc. A digram is a pair of consecutive letters. Considering the words **information** and **informative**, we can calculate the similarity as follows:

Information => in nf fo or rm ma at ti io on

Unique digrams = in nf fo or rm ma at ti io on

Informative => in nf fo or rm ma at ti iv ve

Unique digrams = in nf fo or rm ma at ti iv ve

These terms both have ten unique digrams and have eight similar unique digrams and therefore the similarity measure is computed as follows:

Similarity = 2c/a+b,

where **c** is the similarity coefficient, **a** and **b** are the total digrams respectively. Therefore we have:

(2*8)/ (10+10) *100 = 80%.

This means there is 80% chance that the stem or similarity of "information" and "informative" is in the first eight digrams (Sharma, 2012).

2.5 Stemming algorithms

Stemming can be achieved by conflating various forms of a word into its single represantion. "Stemming is a computational procedure which reduces all words with the same root (or if prefixes are left untouched, the same stem) to a common form, usually by stripping each word of its derivational and inflectional suffixes" (Lovins, 1968). Stemming algorithms have been studied in computer science for a long time. The Lovins Stemmer (Lovins, 1968) was the first ever published stemming algorithm. There are two categories that stemming algorithms can be classified into; namely Rule-based and Statistical approach.

2.5.1 Rule-Based Approach

Porter (1980) introduced this with specific rules for the English language. As mentioned by Ntais (2006), Porters technique is the widely adopted method by other researchers. The algorithm removes suffixes iteratively from a given word reducing it to its stem and it achieved high precision and recall. In a rule based approach language specific rules are encoded and stemming is performed based on these rules.

In this approach various conditions are specified for conflating a word into its derivational stem. Normally, a list of all possible stems is written down and passed to the system and also there are some exceptional rules which are used to handle the exceptional cases (Sharma, 2012). The word to be stemmed is then taken through the table that has words and stems to find a matching variation of a word. This was a good solution in the old time where we did not have much computation capabilities of computers. However, the world is changing and we see improvements in the processing and storage capabilities so as stemming techniques are also adopting the statistical or machine learning approaches. Various researchers showed that statistical Stemmers are good alternatives to rule based Stemmers. Nevertheless some

researchers around the world still use rule based approaches even today for morphologically rich languages.

2.5.2 Statistical/Machine Learning Approach

Most of the researchers believe that statistical stemming approach is the most effective and popular approach in information retrieval. This is because one cannot be exhaustive in listing all the rules, some rules may be missed and also languages have some inherent characteristics that can be represented statistically. Statistical stemming approach employs statistical information and algorithms from a corpus of a language in hand to learn the morphology of words. This gives statistical Stemmers an advantage because they do not need linguists. John Goldsmith did most of the prominent work in this area. He proposed an algorithm to model morphological variants of European languages using an unsupervised approach (Goldsmith, 2001). Another statistical stemming approach was the one by Melucci and Orio (2003) to build a stemming algorithm based on the Hidden Markov Model (HMM). Their design does not need a prior linguistic knowledge or a training set that is manually created. It uses unsupervised training that can be performed at indexing time.

The Hidden Markov Model is a very popular machine learning algorithm. HMMs are defined as finite state automata with transitions defined by probability functions. The probability of each path can be computed and therefore it is possible to find the most probable path in the automata graph using Viterbi coding. Melucci and Orio (2003) suggested that in order to apply HMMs to stemming, a sequence of letters that forms a word can be considered the result of a concatenation of a prefix and a suffix. There are assumptions that can be made in this method and they are as follows:

1. Initial states belong only to the prefix-set –> a word always starts with a prefix.
2. Transitions from states of the suffix-set to states of the prefix-set always have a null probability -> a word can be only a concatenation of a prefix and a suffix.
3. Final states belong to both sets -> a prefix can have a number of different inflections, but it may have also an inflection without any suffix (Melucci & Orio, 2003).

Melucci and Orio (2003) consider three different topologies of HMM in their experiments. The HMM topology as explained by the authors defines the number of states, the labeling of states as

belonging to one of the two sets, the allowable initial and final states and the allowable transitions (Melucci & Orio, 2003). Melucci and Orio (2003) compared their result with that of Porter's algorithm and they found that HMM had a tendency to over-stem words. Melucci & Orio (2003) concluded that the ability of HMM algorithm to work automatically with different languages was beneficial and this was shown in five European languages.

There is another statistical approach known as Yet Another Suffix Stripper (YASS). It is also language independent Stemmer; its performance is comparable to that of Porter's and Lovins Stemmers. The comparison of this algorithm with Porters and Lovins Stemmer was based in both the average precision and the total number of relevant documents retrieved (Sharma, 2012). Majumder *et al.* (2007), describe a clustering-based approach to discover equivalence classes of root words and their morphological variants. They define a set of string distance measures and a lexicon for a given text collection is clustered using distance measures to identify these equivalence classes (Majumder *et al.,* 2007). Their algorithm also provides significant improvements in retrieval performance for French and Bengali, which are resource poor. The following section presents related work.

2.6 Related Work

A lot of research has been undertaken in the area of stemming. Most of these Stemmers were developed for academic purposes. Stemmers are language dependent and therefore each language requires its unique Stemmer. One cannot simply take a Stemmer that was developed for the English language and apply it in isiXhosa documents. This will not work because these are two completely different languages and the developer needs to study and understand the morphology of the language in hand. The following are some of the developed Stemmers for other languages.

2.6.1 Stemming Algorithms for English Language

From the stemming algorithms that have been develped for the English language, we present a few in this Section, which are; Lovins stemming algorithm (1968), Dawson stemming algorithm (1974), Paice/Husk stemming algorithm (1990) and Porter stemming algorithm (1980).

2.6.1.1 Lovins Stemming Algorithm

Lovins Algorithm was designed by Julie Beth Lovins at Massachusetts Institute of Technology in 1968. The algorithm handles individual words; it does not have information about how the words are related or connected with one another (Lovins, 1968). In the first ever published paper on stemming by Lovins (1968), she stated that there are two main principles that are used in the construction of stemming algorithms. These principles are namely *iterative* and *longest-match.* Lovins suggests that these principles should be used together because applying only one may have some drawbacks in the algorithm (Lovins, 1968). Lovins also mentioned that iteration is based on the fact that suffixes are attached to stems in a certain order and therefore there exist order-classes of suffixes. Her article further explains that the longest match principle states that within any given class of endings, if more than one ending provide a match the one with the longest match should be removed first (Lovins, 1968). There is another term described by Lovins as *context sensitive*, which means any matching end or suffix is removed from a stem but the only restriction is that from all the removed suffixes the stem cannot be of length zero. Therefore Lovins developed a single pass, context sensitive and longest match Stemmer (Lovins, 1968). The algorithm checks through the given endings or possible suffixes and searches for the ending that has the longest match with the given word and remove that ending. The algorithm considers 294 endings and 29 conditions that state how each ending should be removed and therefore each ending was linked in one of the 29 conditions. In addition Lovins had 35 transformation rules that allowed for exceptions to be transformed to other endings (Lovins, 1968). If a word is being stemmed and a satisfying condition for an ending is found that ending is removed. Lovins explained that the algorithm is fast but it missed some endings because it was influenced by the technical vocabulary used by the author (Lovins, 1968). Figure 2.2 presents detailed steps of the whole algorithm:

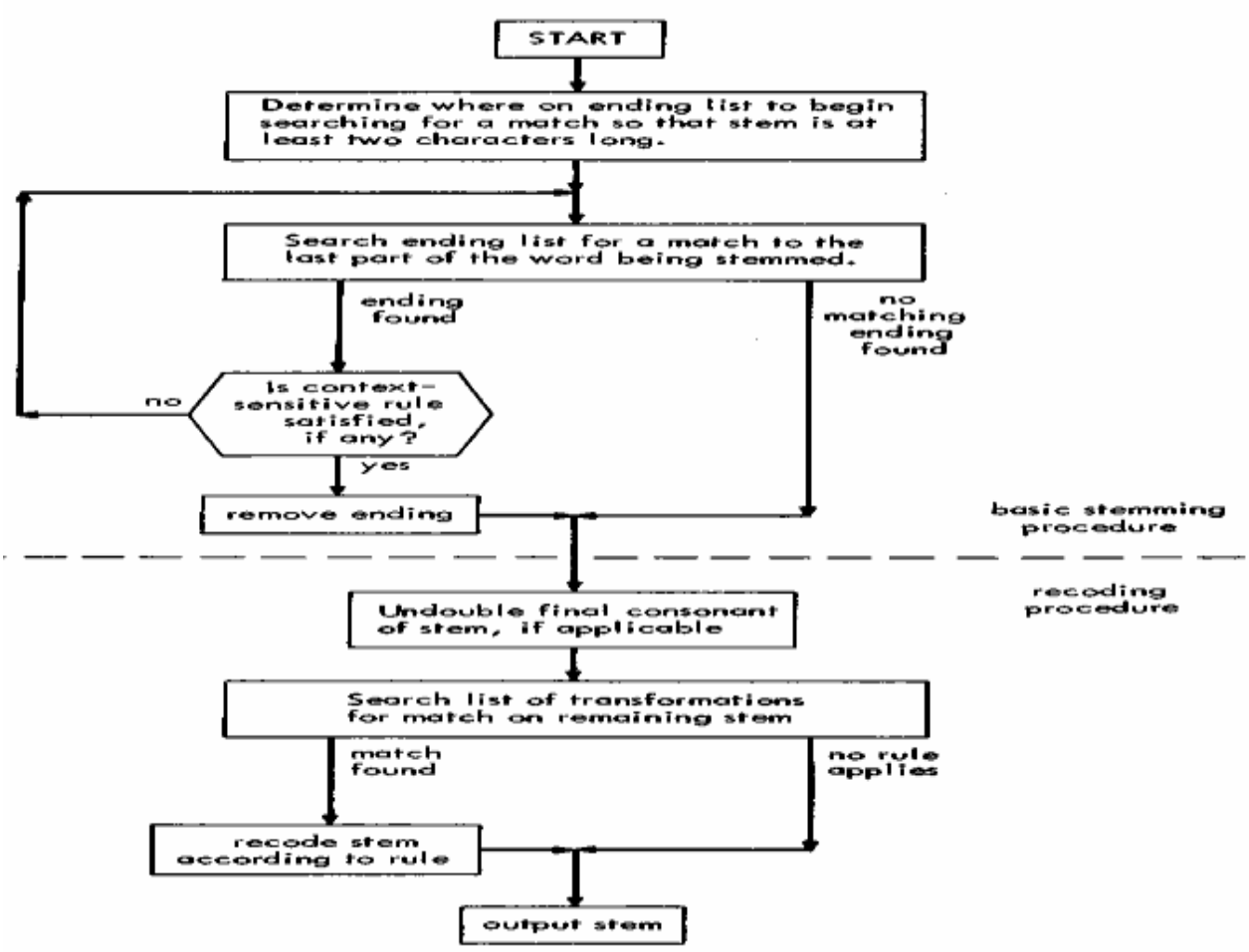

Figure 2.2: Lovins Stemming Algorithm (Lovins, 1968).

Lovins Stemmer cannot be used in its totality in the development of the isiXhosa Stemmer. The Stemmer is for a different language and the prefixes are totally different. However we took some techniques and as such the two algorithms are similar in the fact that if more than one ending or prefix provides a match, the one with the longest match is removed. Lovins Stemmer and the isiXhosa Stemmer have a common principle, that a stem cannot be of the lentgth zero and therefore a noun is only passed once in the Stemmer for affix removal. For both algorithms the list of affixes is given.

2.6.1.2 Dawson Stemming Algorithm

Dawson stemming algorithm is a complex linguistically targeted Stemmer that makes use of iterative longest match approach and is based on the one that was developed by Lovins (1968). This algorithm can also be considered as an improvement of the Lovins approach. Dawson

mentioned that his algorithm has the most comprehensive list of suffixes along with transformation rules. At the beginning Dawson used a list that had 260 suffixes with the associated removal condition codes taken from the Lovins algorithm (Dawson, 1974). Dawson evaluated and corrected the list given by Lovins and he ended up with 1200 suffixes. The suffixes were stored in reversed order indexed by their length and last letter; this was done to avoid the problem of storage and processing time (Dawson, 1974). Dawson used an extension of the partial matching procedure that is also defined in Lovins algorithm instead of the recoding technique.

As the Dawson Stemmer is an improvement of the Lovins Stemmer, it has some similarities and differences with the isiXhosa Stemmer. The isiXhosa Stemmer is developed for nouns and therefore the list of prefixes and suffixes that are present are the ones that are only found in the nouns of the isiXhosa language. IsiXhosa Stemmer has 2 endings as compared to 1200 of the Dawson Stemmer and 40 prefixes all found in the nouns. Most of these English Stemmers are suffix removers, they only consider suffixes and this is different from the isiXhosa Stemmer where we developed both prefix and suffix remover. This is because isiXhosa nouns are classified into morphological classes with different prefixes for singular and plural, therefore making prefix removal an important aspect of this study.

2.6.1.3 Paice/Husk Stemming Algorithm

This Stemmer was developed by Chris Paice and Gareth Husk in Lancaster University in the late 1980s. It was first implemented using Pascal programming language and later an ANSI C and Java version came out. This is an iterative stemming algorithm, with a separate rule file which is first read into an array or list (Paice & Husk, 1987). The file is divided into a series of sections and each section corresponds to a letter of alphabet. The Stemmer uses a single table of rules, each of which may specify the removal or replacement of an ending. The replacement technique is used to avoid the problem of spelling exceptions by substituting the endings rather than simply removing them (Paice & Husk, 1987). This means that in the Paice/Husk stemming algorithm there is no recoding or matching required. The rules in this algorithm are indexed by the last letter of the ending. The algorithm has four main steps, which are:

1. Select relevant section; inspect the letter of the term and, if present, consider the first rule of the relevant section of the rule table.

2. Check applicability of rule; if final letters of term do not match rule, or intact settings are violated or acceptability conditions are not satisfied go to stage 4.
3. Apply rule; remove or reform ending as required and then check termination symbol, and either terminate or return to stage 1.
4. Look for another rule; move to the next rule in table, if the section letter has changed then terminate, else go to stage 2.

These steps are shown in a flowchart in Figure 2.3. The Stemmer is said to be easily implemented and remained efficient but is also known to be very strong and aggressive, it over-stem.

The isiXhosa Stemmer have its rules embedded within the code and in Paice/Husk Stemmer there is a separate file for the rules which it first reads into an array. Paice/Husk Stemmer uses a replacement technique whereas the isiXhosa Stemmer uses a removal technique. IsiXhosa Stemmer rules are arranged according to noun class prefixes, starting from the first noun class up to the last noun class. Paice/Husk Stemmer has its rules indexed by the last letter of the ending instead. Due to differences in the isiXhosa and English language Paice/Husk Stemmer cannot be directly applied to develop isiXhosa Stemmer.

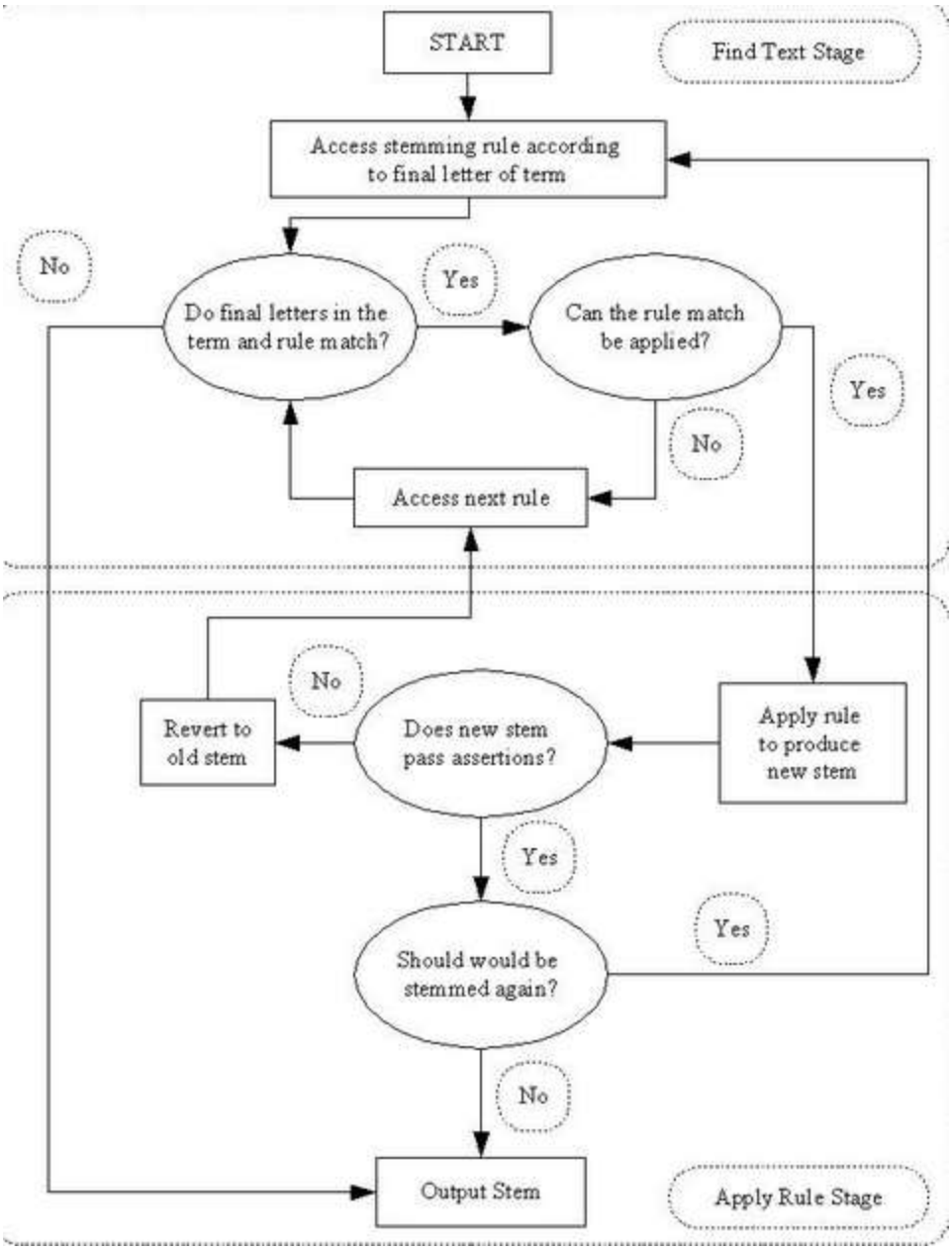

Figure 2.3: Paice/ Husk Stemming Algorithm (Paice & Husk, 1987).

2.6.1.4 Porter's Stemming Algorithm

This algorithm was developed by Martin Porter at the University of Cambridge in 1980 (Porter, 1980). It is a conflation Stemmer consisting five successive steps of word transformation. Each step consists of a set of rules in the form <condition> <suffix> - > <new suffix> (Porter, 1980). The steps are shown in Figure 2.4 below.

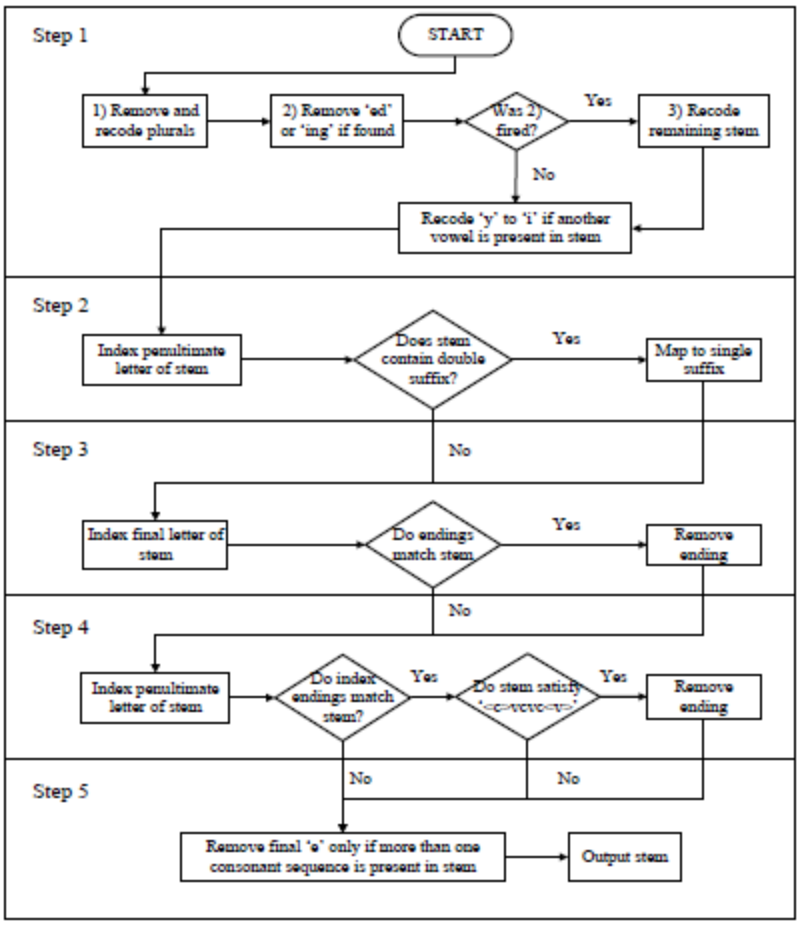

Figure 2.4: Porter's Stemming Algorithm (Porter, 1980).

It is based on the idea that approximately 1200 suffixes in the English language are made up of a combination of smaller and simpler suffixes (Porter, 1980). The author states that within each step, if a suffix rule matches the word, then the conditions attached to that rule are tested on the resulting stem, tested to check if that suffix was removed in the way defined by the rule. The process continues until either a rule from that step fires and control passes to the next step. The resultant stem returned has been passed from step five in Figure 2.4.

In general, the isiXhosa Stemmer and Porter Stemmer are similar in the sense that their rules are arrangend in steps. Within each rule there is a condition and next to the condition are prefixes and suffixes that need to be checked against that condition and removed if the condition is satisfied. The differences lie in the fact that isiXhosa Stemmer is a single pass Stemmer and in Porter it is an iterative process.

2.6.2 Stemming Algorithms for Other Languages

Different Stemmers have been developed for many languages across the globe. The Stemmers that we are reviewing here include Stemmers written for languages such as: Greek, Bengali, Malay and many other languages. The Stemmers are generally customized for each specific language. Their organization and design requires linguistics expertise in the language.

2.6.2.1 Greek Stemming Algorithm

This algorithm was developed by Georgios Ntais in 2006 at Stockholm University in Sweden. In his thesis he presented a stemming system for the Greek language. The system takes a word as input and removes its inflectional suffix according to a rule based algorithm. Ntais developed this algorithm according to the grammatical rules of the Greek language, as they are described in Triantafyllidis grammar 1941 for the Modern Greek language (Ntais, 2006). The algorithm is developed based on the Porter's algorithm. Ntais (2006) mentioned that one has to make some assumptions in order to produce a better working stemming algorithm. Ntais (2006) therefore used capital letters for the system, trying to solve the problem of the "moving" tone-mark on the stems of the Greek words. Ntais (2006) explained that using capital letters in the Greek language, some words may be pronounced in different ways with different meanings each time. Ntais (2006) also explained that prefixes in Greek may change the meaning of the word radically and sometimes the semantics. Therefore in developing this system Ntais (2006) decided not to consider prefix removal. Ntais (2006) mentions that there are some cases of allomorphy in the Greek language, where the verbs starting with a consonant take the letter "ε" as prefix on the past tense. He mentions that for words presented in past tense the stem changes formation and that is why there are two stems for every verb. Ntais also mentioned the Greek language is rich in derivative words, meaning there can be many words coming from the same stem, but an accurate Greek Stemmer should deal with both derivational and inflectional endings. The Stemmer developed by Ntais (2006) was not the first Stemmer that was produced for the Greek language. In 2001, Tambouratzis and Carayannis had developed a system that performs an automated morphological categorization of Greek words extracted from a corpus. This system was developed for Institute for Language and Speech Processing in Greece (Ntais, 2006). The algorithm was designed to use a rule-based iterative matching and masking approach. Its aim is to perform the segmentation of a given set of words into stems and endings in an automated manner. Tambouratzis & Carayannis (2001) made an assumption that each word consists of a

stem part and an ending part excluding the compound words. Their system is presented in Figure 2.5 and is called AMP.

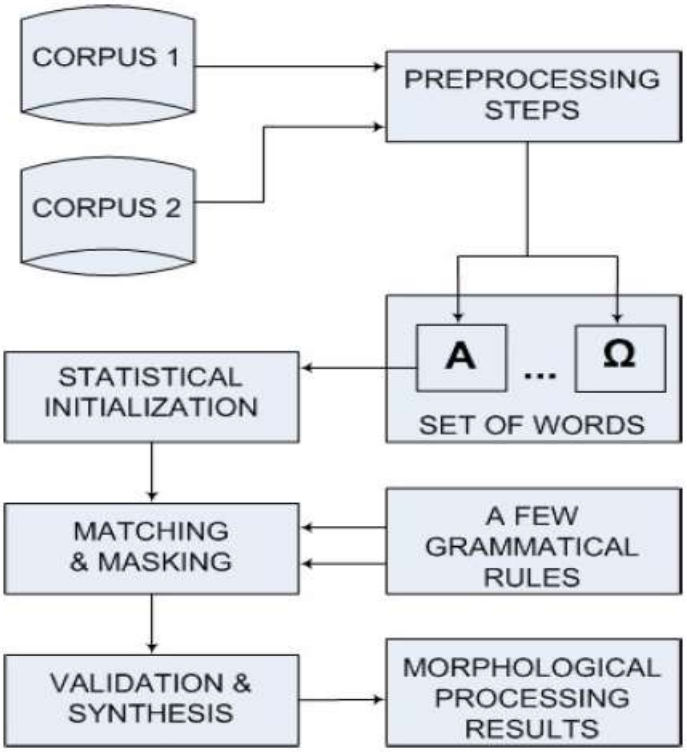

Figure 2.5: Automated Morphological Processor (AMP) Overview (Fotinea *et al.*, 2001).

The AMP system performs successful stemming for Greek words and it has a stemming accuracy of 95%. In (Fotinea *et al.*, 2001), the authors have highlighted that the few grammatical rules that follow the matching and masking process are not enough to consider as complete stemming algorithm for the Greek language. The isiXhosa Stemmer may not be compared to the Greek Stemmer because of the differences in the morphology of the two languages. Both Stemmers made use of the well known Porter's Stemmer but because the isiXhosa Stemmer is developed for nouns only and therefore this lead to variations between the two. In the Greek Stemmer prefixes were not even considerd and in the isiXhosa Stemmer prefixes were considerd and they formerd major part of this study as the noun classes are differentiated by their prefixes. In developing the isiXhosa Stemmer we only used small letters and for the Greek Stemmer only capital letters were used. However, both algorithms are rule based.

2.6.2.2 Stemming Algorithm for Bengali

Bengali is an Eastern Indo-Aryan language and it is native to the region of Eastern South Asia known as Bengal. It is one of the most spoken languages, ranked seventh in the world. It is highly inflected language with relatively free word order (Islam *et al*., 2007). Its nouns are inflected for the following cases; nominative, genetive, objective and locative. Verbs are either finite or non-finite. By non-finite verbs we mean the type of verbs that are not inflected for person or tense. Finite verbs are fully inflected for tense and person. Similar to any other language, Bengali words contain both prefixes and suffixes. When developing the stemming algorithm for Bengali language Islam *et al* (2007) came up with a suffix stripper (Islam *et al.*, 2007). Islam *et al* (2007) mentioned that when a word contains more than one suffix, suffixes glue to each other to form longer suffixes and they need to be removed iteratively. In their algorithm they found 22 suffixes for nouns, 72 for verbs and 8 for adjectives.

The algorithm is designed in a way that suffixes are listed separately and when a particular word is stemmed it goes through the list and compares the suffixes then removes the longest match. In their algorithm the suffixes are ordered according to length so that the longest suffix is at the top of the list (Islam *et al*., 2007). The Stemmer was used for a spelling checker in the text editor of Bengali, and the authors mentioned that the use of a Stemmer showed significant improvement in the spelling checker's performance (Islam *et al*., 2007).

The Bengali Stemmer and the isiXhosa Stemmer are both longest match but the difference lie with the fact that for Bengali language they developed a suffix stripper and in the case of isiXhosa language we developed a prefix and a suffix stripper.

2.6.2.3 Stemming Algorithm for Malay Words

Malay is spoken in South East Asia, mostly in Malaysia, Indonesia and Brunei. The language is spoken by more than 150 million people. The stemming algorithm for Malay language was originally developed by Othman in 1993. The algorithm was developed using the rule-based approach that makes updating the language rules easier. The author mentioned that this type of approach reduced the stemming process ten times slower than Porter's stemming algorithm. The way the algorithm worked was simply in a way that affixes are removed through a process of matching the affixes in the rules to that of input word (Othman, 1993). The general flow of the algorithm is as follows:

Step-l: If there are no more words then stop, otherwise get the next word;

Step-2: If there are no more rules then accept the word as a root word and go to Step-1 ,otherwise get the next rule;

Step-3: Check the given pattern of the rule with the word: If the system finds a match, apply the rule to the word to get a stem;

Step-4: Check the stem against the dictionary; perform any necessary recoding and recheck the dictionary;

Step-5: If the stem appears in the dictionary, then this stem is the root of the word and go to Step- 1, otherwise go to Step-2.

There are three characteristics of this algorithm; the algorithm can over-stem, the operations precise mode of the algorithm depends on the order of the rules and lastly the total number of rules used. Othman (1993) applied the rules for four affix classes; first being the prefix-suffix rules, then the prefix rules, then the suffixes rules and lastly infix rules.

The Malay Stemmer and the isiXhosa Stemmer are developed for the same purpose, they both remove prefixes and suffixes. The Malay Stemmer differs because it also removes infixes and we have no infixes in the isiXhosa Stemmer. They are rule based Stemmers, that take the input word and checks it against the list of rules and conditions. Due to morphological complexity and differences in the two languages, they cannot use the same rules and conditions. This means that any language that needs a Stemmer, it must be developed from scratch because every language needs to be studied carefully and its rules applied correctly.

2.7 Evaluation Methods for Stemming Algorithms

There are different techniques or methodologies that are used to evaluate the performance of Stemmers. The Paice method, the manual method and vocabulary reduction are the most widely used Stemmer evaluation methods (Abedo, 2012). In the vocabulary reduction method, it is where the number of words in the corpus is divided by the number of stems generated excluding repeated stems. In the manual method, a human expert/linguist is called to decide if the system produced the correct stem for every word. That is, in this method a linguist checks the number of correct results, errors due to over-stemming and under-stemming. The Paice method was described by Paice (Paice, 1996). This method uses error counting, two types of errors in particular; under-stemming and over-stemming. The method returns the value of an Under-

stemming Index (UI), calculated from the Conflation Index (CI). CI is defined as the proportion of the equivalent word pairs which were successfully grouped into the same stem and therefore, UI = 1-CI. The value of an over-stemming is calculated as Over-stemming Index (OI) and Distinctive Index (DI). Where the DI is the proportion of non-equivalent word pairs which remained distinct after stemming and therefore OI = 1-DI (Paice, 1996).

2.8 Conclusion

In this Chapter, we discussed about the natural language processing techniques and their applications. The Chapter basically, presented the literature review of the natural language processing as a bigger picture. Stemming, the main focus of this study, also falls under this bigger branch of natural language processing and therefore stemming algorithms were also presented. Different algorithms are explained in detail including the different conflation techniques. Then an overview of English Stemmers was given and non-English languages were also presented. The Chapter concluded by giving the evaluation methods of stemming. The following Chapter will explain the morphology of the isiXhosa language.

Chapter Three

IsiXhosa Language

3. IsiXhosa Language

3.1 Introduction

The objective of this Chapter is to discuss the morphology of the isiXhosa language, paying closer attention to the nouns as that is the scopet of this work. The Chapter begins with the brief history of the language, where it originated, how it is divided amongst those who speak it (dialects), the population of speakers in South Africa and how each dialect is related to the other. The Chapter furthermore discuss how the words especially nouns are formed, how the affixes are attached to the root words so that they give a more significant meaning.

3.2 History of the isiXhosa Language

The word Xhosa is from the Khoisan language and in that language it means "The Angry Men". Those people who speak this language typically form part of the ethnic group called amaXhosa refers to their language as isiXhosa. AmaXhosa, also known as the Southern or Cape Nguni are composed of numerous groups of people concentrated mainly in the former Transkei, Ciskei, Eastern Cape and Western Cape regions. As with other African languages that are tonal, isiXhosa also has tone as one of its important elements (Pascoe & Smouse, 2012). IsiXhosa can be described as more agglutinative (joining/ connective morphemes to form words some of which have syntactic value), than tonal. There are tonal differences but not so much as to have tone defined as a governing feature. Tone is important but not over-arching. IsiXhosa, rather, is governed by the noun which dominates the sentence (Pascoe & Smouse, 2012).

IsiXhosa is one of the languages that are known as Nguni languages which include; siSwati, isiNdebele and isiZulu. All these languages share many linguistic features. These Nguni languages also form part of a bigger group of Bantu languages. The isiXhosa language has been grouped into several dialects namely; isiBhaca, isiMpondo, isiMpondomise, isiGcaleka, isiNgqika, isiHlubi, isiXesibe, isiThembu, isiMfengu and isiBomvana. Most of these dialects are found mostly in the former Transkei. The role of African languages in South Africa is complex as their use in education is governed by legislation. IsiXhosa is used as the main language of instructions in many primary schools and some secondary schools in the Eastern Cape Region. It is also studied as a subject.

Approximately 8 million people in South Africa speak isiXhosa as their mother tongue (Census, 2011). It has five simple vowels a, e, i, o and u. The isiXhosa is also known for its clicks. It has three basic click consonants; a dental click with the letter "c [/]", a lateral click with the letter "x [//]" and a palatal click with the letter "q [!]" represented in both Orthographical and Phonetic representation, these are more standardly or generally represented as shown in the International Phonetic Alphabet (IPA) chart (IPA Chart, 1996). IsiXhosa nouns are classified into fifteen morphological classes, with different prefixes for plural and singular. It is an agglutinative language consisting of a list of prefixes and suffixes attached to root words or stems (Pascoe & Smouse, 2012).

3.3 IsiXhosa Orthography

John Bennie (1823) put isiXhosa into writing for the first time. His writing was a disjunctive form and it was a difficult system because it was dependent on how each individual formed syllables out of each word (Saul, 1998). His form of writing was as follows:

In ko mo zon ke ze zi ka- Ti xo: un gum ni ni zo ye na.

"All the cattle belong to God: He is their owner".

In this form of writing it was noticed that in some words where there is supposed to a nasal compound, the nasal consonant is left out. Examples of these words are given by Saul (1998) as, ***go ko*** instead of ***ngoko*** "therefore", ***ga yo*** instead of ***ngayo*** **"**with it" and ***go ku ba*** instead of ***ngokuba*** "because". Saul (1998) further argues that this is due to the fact that to a foreigner this voiced velar nasal compound is not easily audible and therefore Bennie has written according to the way he heard the sounds. The use of hyphen is also noticeable in John Bennie's way of writing.

After thirty years of John Bennie syllabic system of writing there came W.G. Bennie who wiped the syllabic writing style and came up with the conjunctive way of writing (Saul 1998). His way of writing was as follows:

Ngennxa "as a result of"

Ummoya "air"

Inncwadi “a book”

These examples are of those nouns that begin with nasal consonants **m** and **n** where a complete prefix of that class was added, thus doubling the nasal (Saul, 1998). In 1915 Kropf used diacritics in his writing to represent the voicing of sounds in some words. His way of writing was as follows:

Bala “write

Kulu “great”

In 1925 Mqhayi introduced his way of writing some of the sounds without diacritics. He wrote some of those sounds as follows:

Ingqeqesho “discipline”

e-Ncera “at Ncera”

The use of diacritics in the isiXhosa orthography came to an end and it was replaced by the use of the International Phonetic Alphabet symbols (Bennie, 1931).

3.3.1 International Phonetic Alphabet

Brown Adam (2013) defines International Phonetic Alphabet (IPA) as a “set of symbols designed to be used for representing the speech sounds of languages of the world”. It was previously known as the International Phonetic Association from its foundation in 1886. The aim of the IPA is to promote the study of the science of phonetics and the various practical applications of that science (International Phonetic Association (ed), 1999). The IPA is based on the Roman alphabet and the advantage for this is that the Roman alphabet is widely familiar. There are also additional symbols and letters from other sources and these additions are necessary because the variety of sounds in languages is greater than the number of letters in the Roman alphabet (International Phonetic Association (ed), 1999). IPA is used for many purposes, these include its use to form the basis of a writing system for a language, to record a language in linguistic fieldwork and it can be used as a way to show pronunciation in a dictionary (International Phonetic Association (ed), 1999). The copy of the IPA is attached in Figure 3.1.

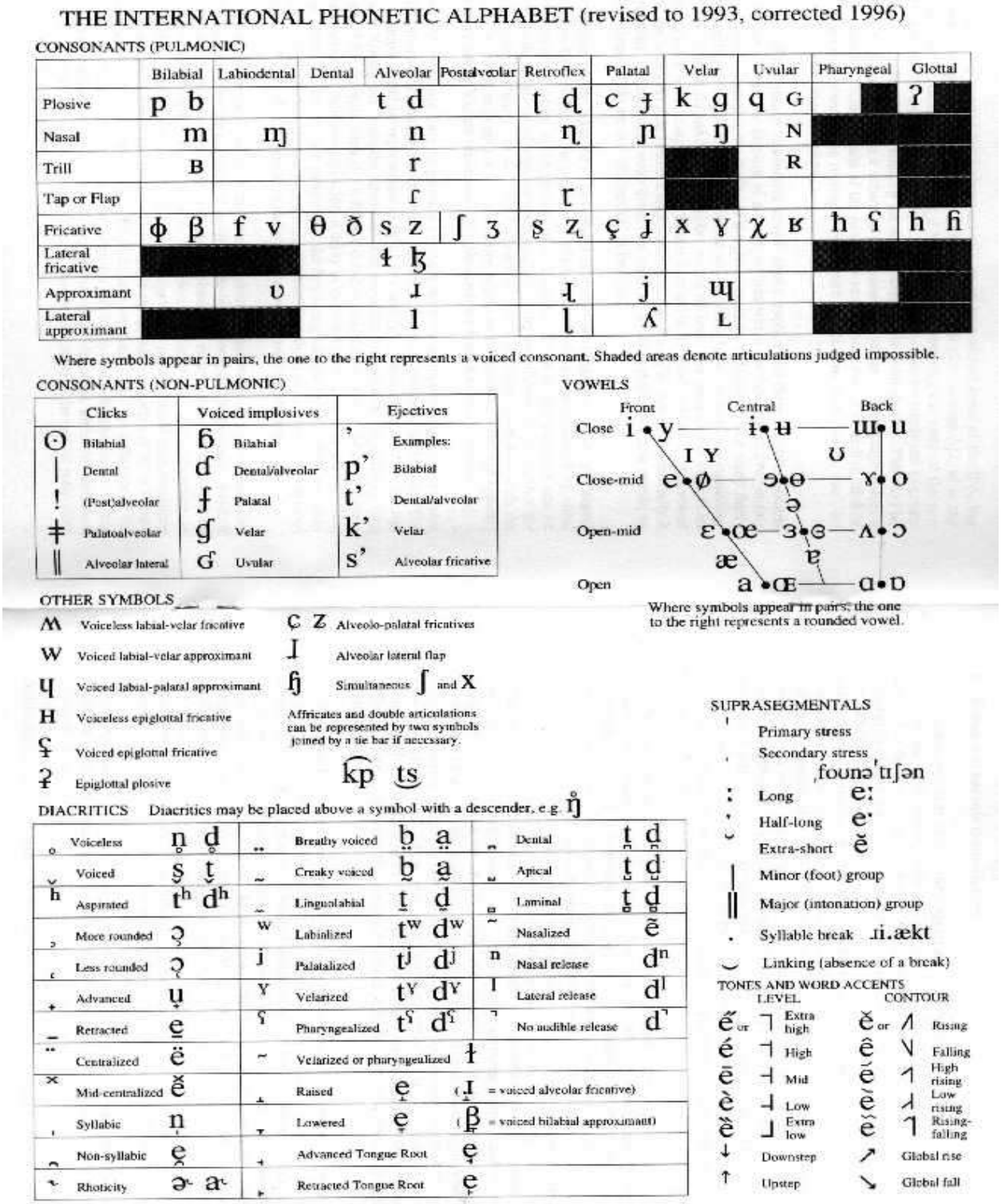

Figure 3.1: International Phonetic Alphabet.

3.4 IsiXhosa Noun Classes

Craig C. G. (1986) suggested that, in linguistics the term noun class refers to a system of categorizing nouns. Therefore isiXhosa just like most Bantu languages has a class system. According to Craig a noun may belong to a given class because of characteristic features of its referent. In the isiXhosa language all nouns belong to one of the thirteen different classes. These classes are usually numbered from 1 to 15 with classes 12 and 13 with the ones higher than 15 missing.

3.4.1 IsiXhosa Noun Class Prefixes

A prefix is a morpheme that is placed at the beginning of a root or stem of the word (Satyo, 1988). Unlike European languages the IsiXhosa nouns are classified according to prefixes. IsiXhosa like many other African languages, especially those categorized as Bantu languages by philologists such as Greenberg and Meinhoff, isiXhosa classifies nouns according to prefixes. All Nguni and Sotho languages do. IsiXhosa has about fifteen noun class prefixes also known in isiXhosa as *Amahlelo* (Satyo, 1988). The first noun class prefix has the prefix **Um-**. The following are examples of the first noun class prefix:

Um*fo* "guy"

Um*thwa* "bushman"

Um*ntu* "person"

Um*oni* "sinner"

Um*embi* "digger"

Following the first noun class prefix is the second noun class prefix and it is also the plural of the first noun class prefix. This is typically how the plurals are rendered in the language, by putting them in a different class. It has the prefix **Aba-, Abe- and Ab**. Following are the examples:

Aba*fo* "guys"

Aba*thwa* "Bushmen"

Aba*ntu* "people"

***Ab**oni* “sinners”

***Abe**mbi* “diggers”

The third noun class prefix has same prefix as the first, **Um-.** The difference is the preceding noun class prefix. For example, in the first noun class its plural is the second noun class but with the third noun class the plural differs from the second noun class. Below are the examples:

***Um**lilo* “fire”

***Um**zi* “house”

***Um**thi* “tree”

Its plural is the fourth noun class prefix which has the prefix **Imi-**. Below are the examples:

***Imi**lilo* “fires”

***Imi**zi* “houses”

***Imi**thi* “trees”

The fifth noun class prefix follows with the prefix **Ili-** for mono-syllabic nouns and just the vowel **i-** for all other nouns. Below are the examples that are mono-syllabic:

***Ili**tye* “stone”

***Ili**wa* “cliff”

***Ili**zwi* “voice”

***Ili**shwa* “misfortune”

Now looking at di-syllabic and polysyllabic nouns

***I**qaqa* “a skunk”

***I**thumba* “a boil”

***I**fokofoko* “a weakling”

I*xaxavithi* “a slothful person”

For these nouns one never sees

the –li-, but their plurals are marked by Ama- of class 6.

The sixth noun class prefix follows the fifth and with the prefix **Ama-**. It is the plural of the fifth noun class. Below are the examples:

Ama*tye* “Stones”

Ama*wa* “cliffs”

Ama*zwi* “voices”

Ama*shwa* “misfortunes”

The seventh noun class prefix has the prefix **Isi-, Is-**. Below are the examples:

Isi*qhamo* “fruit”

Isi*zwe* “nation”

Isi*tiya* “garden”

Isi*baluli* “identifier/ relative (in grammar)”

Is*azi* “intellectual/wiseman”

The eighth noun class prefix has the prefix **Izi-**, **Iz-**. It is the plural of the seventh noun class. Below are the examples:

Izi*qhamo* “fruits”

Izi*zwe* “nations”

Izi*tiya* “gardens”

Izi*baluli* “identifiers/relatives (in grammar)”

***Iz**azi* “intellectuals/wisemen”

The ninth noun class prefix has the prefixes **In-**, **Im-**, **I-.** Below are the examples:

***In**to* “thing”

***Im**babala* “springbok”

***I**tafile* “table’

Following is the tenth noun class prefix. It is the plural of the ninth noun class. It has the prefixes **Izin-**, **Izim-**, **Ii-**, **Iin-**, and **Iim-**. Below are the examples:

***Izin**to* “things”

***Iim**babala* “springboks”

***Ii**tafile* “tables”

***Izim**vo* “opinions”

***Iin**kawu* “monkey’s”

The eleventh noun class prefix has the following prefixes: **Ulu-**, **Ulw-**, **Ul-**, **U-**. Most of the plural form of this noun class falls under the tenth noun class prefix. There are also those plural that do not fall under the tenth class noun prefix. Below are the examples:

***Ulu**thi* “stick”

***Ulu**tho* “something”

***Ulw**azi* “knowledge”

***Ul**onwabo* “happiness”

***U**phawu* “a sign”

The twelfth and thirteen noun class prefixes do not exist because in Proto-Bantu these two contained diminutives, which in isiXhosa are signified by suffixes and not prefixes and in some other Bantu languages that still do have noun classes 12 and 13, and so we jump straight to the

fourteenth noun class prefix. It has the following prefixes: **Ubu-**, **Ub-**, **Utyw-**, **Uty- and U-**. Below are the examples:

Ubu*ciko* "eloquence/ability to sing well"

Utyw*ala* "alcohol"

Uty*ani* "grass'

U*boya* "wool/body hair"

Ub*usi* "honey"

The fifteenth noun class prefix has the following prefixes: **Uku-**, **Ukw-**, **Uk-**. The prefix that is most welcomed to be used in this noun class is **Uku-**. Below are the examples:

Uku*tya* "food"

Uku*phanda* "investigate"

Ukw*aba* "sharing"

Uk*ona* "to sin"

Uk*osa* "braai/roast"

There are other two noun class prefixes which are noun class prefix 1(a) and 2(a). Noun class 2(a) is the plural of 1(a). The prefix in these two classes is missing; we only have the augment "*iceba*". These two are mostly used to refer to nouns that refer to human beings and a few folktale characters that are animals. Below are the examples:

1(a)	2(a)
umama "mother"	*oomama* "mothers"
unovenkile "shopkeeper"	*oonovenkile*"shopkeepers"
udyakalashe "jackal"	*oodyakalashe* "jackals"

The noun class prefixes are very important in this research as they are the prefixes that are removed from a noun to form stems. Therefore all the noun class prefixes listed in this section are the ones that are embedded on the rules of the Stemmer.

3.4 The Role of the Prefix in IsiXhosa Parts of Speech

A prefix plays a very important role in determining the part of speech in the isiXhosa language. It is also the role of the prefix to change the part of speech in the isiXhosa words. The prefix also plays a significant role in highlighting whether a word is in singular or plural form. E.g., ***Um****zi* "House" = ***Imi****zi* "Houses", ***In****ja* "Dog" = ***Izi****nja* "Dogs" and ***Um****thwa* "Bushman" = ***Aba****thwa* "Bushmen". Looking at the examples below gives a clear picture of the role that the prefix plays in the part of speech in the language.

Uku*dala* "to create" (Infinitive Verb)

Isi*dala* "ex or a prolonged period" (Noun)

En*dala* "the old one" (adjective)

3.5 The Role of Prefix in Noun formation

Each class has its set of prefixes and in the nominative case of the word the prefix consists of two parts; the initial vowel known as the augment also known as the pre-prefix and the actual prefix also called the prefix proper (Satyo, 1988). Let us look at the following examples of noun formations:

Inja "a dog" => *i- iceba* "augment"

-n- isisekelo "base"

-in- isimaphambili "prefix"

-ja- isiqu "stem"

Izilambi "the poor" => *i- iceba* "augment"

-zi- isisekelo "base"

izi- isimaphambili "prefix"

-lambi- isiqu “stem”

In the cases of Indo-European languages, nouns can occur in the number of different forms in a sentence and this is also the case in this language but the similarity to Indo-European languages is superficial. These cases can be called Vocative, Locative, Predicative and Negative Predicative.

Table 3.1: Forms of Nouns.

Word	Case	Meaning
Abantu	Nominal	People
Bantu	Vocative	People
Ebantwini/ Kubantu	Locative	To/From the people
Ngabantu	Predicative/ Copulative	It is people/ they are people
Asibantu/ Asingobantu (ayingobantu)	Negative Predicative	It is not people

3.5.1 Vocative

The vocative of a noun is formed by ***ushiyo lweceba*** “dropping the augment”. The augment can be dropped in many different situations (Bourquin, 1946). For the purpose of this study we will not go into details about dropping the augment.

3.5.2 Locatives

Locatives can be formed in many ways, but the most common one is to replace the augment “*iceba*” by **e-** and replace the final vowel by **ini-**. Below are examples of such nouns.

Intaba “mountain” => *Entabeni* “at the mountain”

Umlambo “river” => *Emlanjeni* “at the river”

3.5.3 Copulatives/ Predicative

Copulatives can be formed mostly from weak classes. By weak classes we mean classes that their prefix contains a nasal, e.g., class 1,3,4,6 and 9. It is formed in classes that starts with **u-** and **a-** by prefixing **ng-**, with exceptional case in class 9 where the prefix becomes **y-**. Strong classes are classes that contain plosives in their prefixes. In these classes a copulative can be

formed by repeating the plosive of the prefix (Jokweni, 1997). Let us consider the following examples to get a clear picture.

(Class 1) *Umntwana* "child" => *Ngumntwana* "it (he/she) is a child"

(Class 9) *Itafile* "table" => *Yitafile* "it is a table"

(Class 5) *Ilitye* "stone" => *Lilitye* "it is a stone"

3.5.4 Negative Copulatives/ Predicative

These copulatives have their own prefixes and are shown in the examples below.

(Class 6) *Ngamakhwenkwe* "they are boys" => *Asingomakhwenkwe*"they are no boys"

(Class 2) *Babantwana* "they are children" => *Asingobantwana* "they are no children"

3.6 IsiXhosa Roots

The root in the isiXhosa language is the one that carries the meaning of the word. By putting a prefix and a suffix it is then where we get the clear meaning of the word. They also help in alternating the word into different parts of speech (Mtuze *et al.,* 1987). Let us look at the following examples before moving forward. In the example we illustrate the formation of the word ***Ixhegokazi*:**

I- : Prefix

-xheg- : Root

-o- : final vowel

-kazi: Feminine Suffix

As mentioned above the root carries the true meaning of the word and this meaning can only be altered if a prefix or a suffix is inserted into the root. The new meaning formed after the prefix or suffix is inserted is determined by the type of prefix used. Let us now look at the following root, ***-theth-*** . This root can form a number of words depending on the type of prefix or suffix inserted.

- *Ukuthetha* "to speak" (infinitive verb)
- *Isithethi* "speaker' (noun)

- *Othethayo* “the speaker”(relative)
- *Intetho* “the talk” (noun)

Another example showing alterations in the parts of speech in a certain root because of a prefix or suffix is the root ***–hamb-.***

- *Ukuhamba* “to walk” (infinitive verb)
- *Umhambi* “the visitor” (noun)
- *Ohambayo* “the walker” (relative)
- *Uhambo* “the journey” (noun)

3.7 IsiXhosa Suffixes

Suffixes are attached at the end of a root word, they are also known as *Izimamva* in isiXhosa. They are given this name in isiXhosa because they are found at the end of a root or the whole word (Mtuze *et al.*, 1987). If a suffix is a vowel it is given the name *“isigqibelo*”. In the isiXhosa language all the verbs and nouns end with a vowel. In Table 3.2, we illustrate how the suffix or a final vowel is attached to the root to form a stem.

Table 3.2: Word formation in isiXhosa.

***Isimaphambili*/ prefix**	**Root/*ingcambu***	**Suffix/*isimamva***	**Final Vowel/ Isigqibelo**	**Stems**
I-	-sel-	-	-a	-sela “thief”
Um-	-thung-	-	-i	-thungi “tailor”
Im-	-bonis-	-el-	-o	-bonisel “to see”
Um-	-bon-	-is-	-o	-boniso “show”
U-	-thand-	-	-o	-thando “love”

The isiXhosa language has what is known as nominal suffixes, and these denote the Feminine, Augmentative and Diminutive. These suffixes have the following properties; “*Kazi*” = Feminine or Augmentative and “*Ana*” =Augmentative (Du Plessis and Visser, 1992). The following examples illustrate the use of these suffixes. *Nkosi* “chief” + *-Kazi* “feminine” = *Nkosikazi* “female chief”. *Umthi* “tree” + *-Kazi* “augmentative” = *Umthikazi* “big tree”. For Diminutive

suffixes we show the following examples. *Indoda* “man” + *-Ana* “diminutive” = *Indodana* “young man”. *Incwadi* “book” + *-Ana* “diminutive” = *Incwadana* “small book”.

3.8 Conclusion

In this Chapter the history of the isiXhosa language was presented as well as the form of writing that was used before. The introduction of isiXhosa to the use of IPA was also discussed. The Chapter thereafter presented a detailed discussion of the noun formation as the core objective of this research was to develop a noun Stemmer. The importance of prefixes and roots in isiXhosa was discussed. The Chapter concludes with a detailed discussion of the suffixes in isiXhosa language. The following Chapter presents the methodology and design that was followed in the development of the Stemmer for the isiXhosa language.

Chapter Four

System Design

4. System Design

4.1 Introduction

The objective of this Chapter is to present the design of the system that we came up with in the development and implementation of the Stemmer for the isiXhosa language. The Chapter will therefore explain all the steps that were followed to come up with the Stemmer. The architecture and the system back-end are also discussed. The user's role in the system is also shown. The Chapter concludes by presenting the rules used to develop the Stemmer.

4.2 Use Case Diagrams

4.2.1 User Roles

The Stemmer for the isiXhosa language is designed and can be accessed by different types of users through the use of an IR system, which are the system administrator and the general users.

4.2.2 Use Cases for System Administrator

The system administrator has the privilege to update the system in cases where new developments have been made in the language. Such cases include the addition of new prefixes, addition of new suffixes and removal of the unwanted ones.

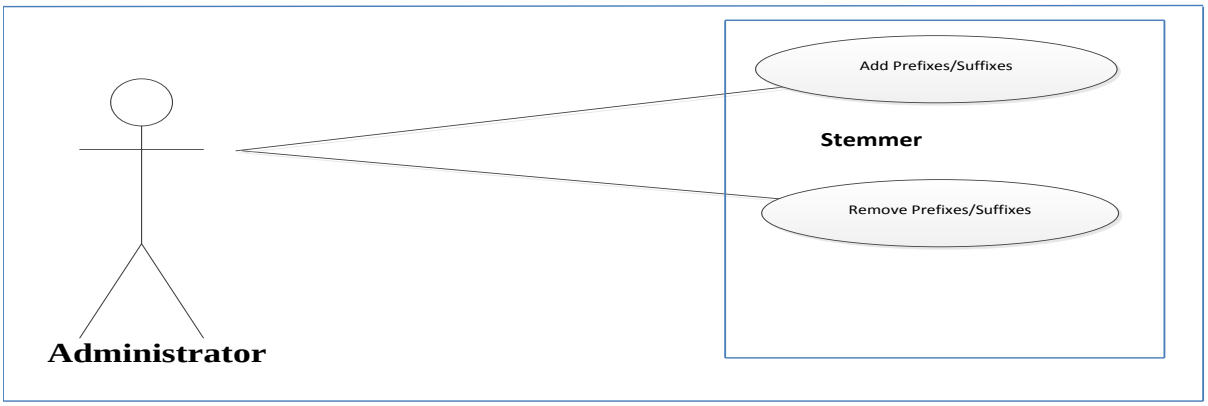

Figure 4.1: Administrator Use Case Diagram

4.2.3 Use Cases for General Users

The general users have no privilege of updating the system; they can only use it by supplying a list of nouns that they want stems of, without the knowledge of what is happening at the back-end of the system. However, for other languages such as English and Greek, when they were developing Stemmers they had two different types of systems with regard to interface. Some of them developed their systems in such a way that their system read words from a file then print stems and in other cases a simple interface was developed where a user type a word on the input box then a stem is printed below in a textfield. For our Stemmer we also followed this trend where we also developed a system similar to that of Porter known as "Snow ball', where the general users can access the interface entering words to get stems.

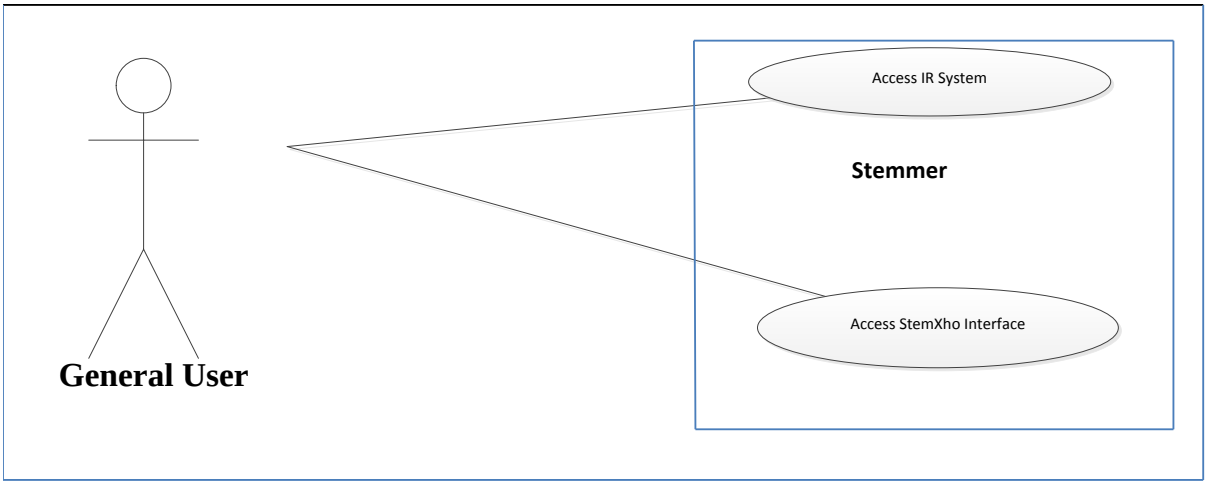

Figure 4.2: General User Use Case Diagram

4.3 The isiXhosa Stemming Algorithm

4.3.1 System Flow Diagram

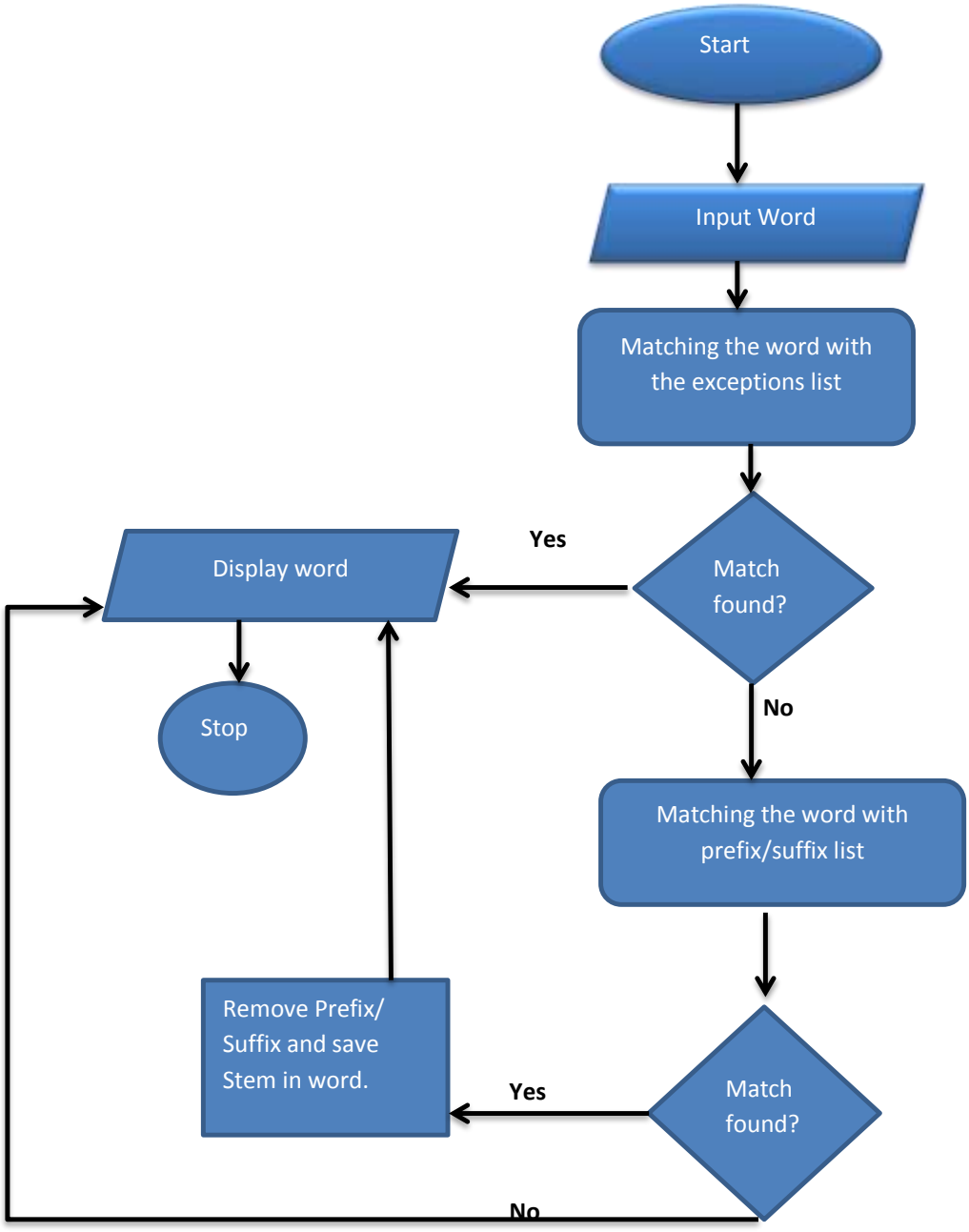

Figure 4.3: Proposed flow chart for the IsiXhosa Stemmer.

The main aspect of this algorithm is to filter a word through the list of exceptions and then if there are no exceptions found, it proceeds to remove prefixes and suffixes which contain all

possible endings. In general, 16 isiXhosa noun classes have been studied and each of these noun classes may contain more than one prefixes. In general, a total of 40 prefixes have been used in the design of this algorithm. These prefixes include the prefixes that are commonly found in the Vocative, Locative, Predicative and Negative predicative forms of a noun. All of these noun classes are detailed in Chapter Three. The isiXhosa Stemmer that was designed used a type of approach that remove prefixes and suffixes and therefore cannot distinguish between word forms or parts of speech. Figure 4.3 shows the basic steps we followed to design the Stemmer. The proposed algorithm is the longest match and there were exceptions to be considered first, such as when a term has three or fewer characters, such cases cannot be stemmed. From our observation we noticed that people's names in isiXhosa do not have affixes that are removed to get stems. People's names sometimes start with a vowel and in developing this Stemmer we included this exception and the vowel is removed accordingly. If a person's name contains the initial vowel, it's dropped. For example: ***u-****Nkosinathi = Nkosinathi*. Then after the exception list is checked the Stemmer removes the prefixes first then suffixes, the reason for this choice is because some of the isiXhosa nouns do not contain suffixes that need to be removed when we want to match related terms. We need to check if they contain suffixes or not. These terms that do not have affixes, may have suffixes and they would be removed for other purposes other than matching related terms for information retreival purposes. Such purposes include noun formation and nouns that build from other nouns. There were cases where different words have the same stem and yet their meaning is completely different, but these terms were stemmed the same way because the algorithm cannot differentiate between terms or it cannot tell the forms in which words are in. Examples of such nouns are; ***i-****hlobo* "summer", ***um-****hlobo* "friend", these two terms have the same stem but the meaning is completely different. The proposed algorithm is presented in Figure 4.3. This algorithm has rules embedded in it, and these rules determine which suffixes or prefixes are removed for a certain word. For example, the minimum number of characters that a word should have in order to be stemmed is four. In the set of rules it is assumed that the longest noun that the algorithm can take is four characters. This is taken into consideration so as to accommodate mono-syllabic stems like, *Fa* (die) and *Nja* (dog).

4.3.2 System Back-end Design

The system back-end is used to support the front-end services and allow users who interact with the application on the front-end system to make requests on the back-end system. The back-end design of the isiXhosa Stemmer is used to control the interaction of users with the IR systems; this includes taking keywords from their queries and stripping off their affixes.

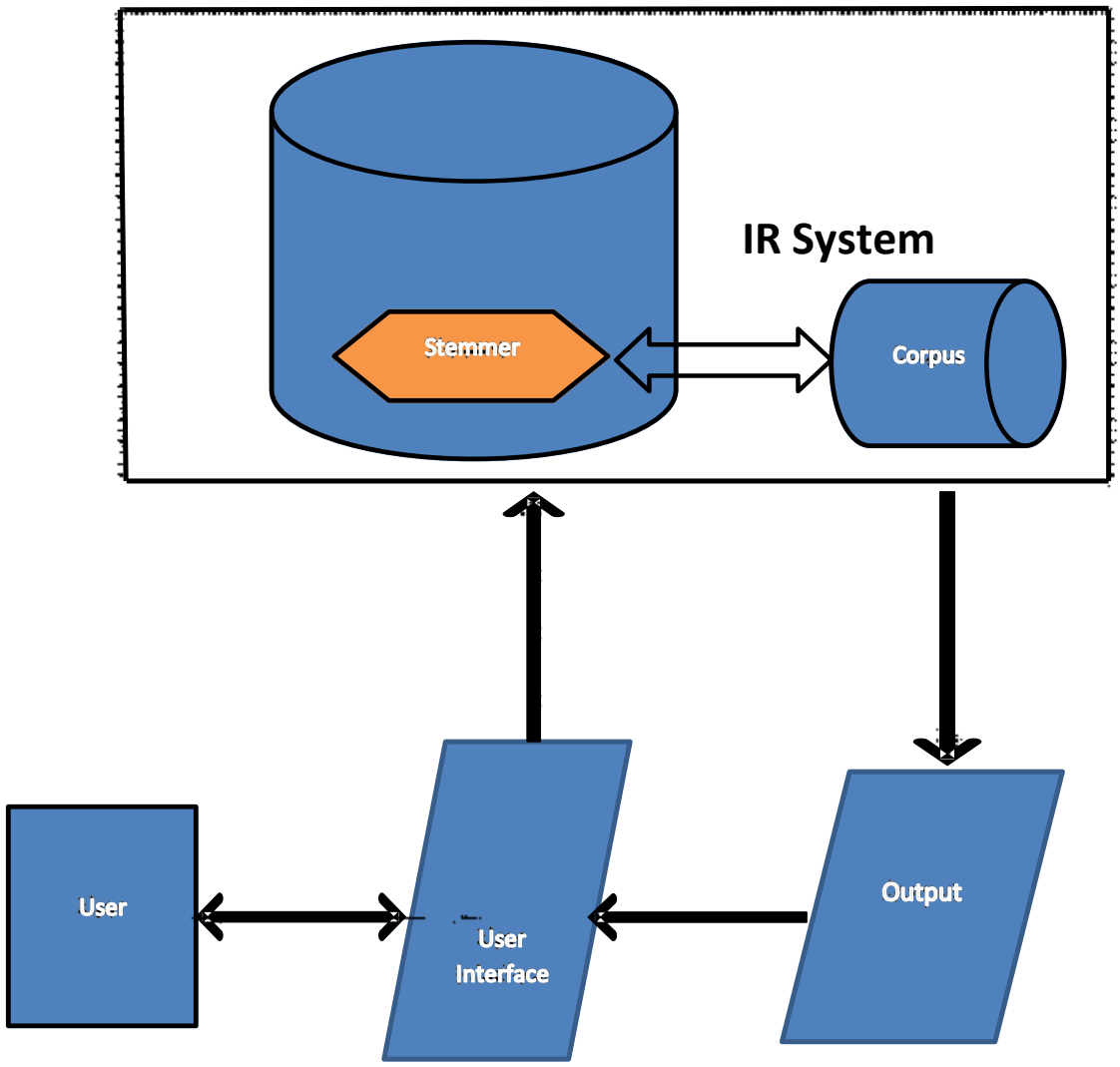

Figure 4.4: Front-end and Back-end architecture

Figure 4.4, above, shows the back-end and front-end design of the system. Figure 4.4, depicts the communication of the Stemmer with the User Interface. As shown in the Figure 4.4 a user types in a query in the User Interface, and the User Interface then uses the Stemmer to get the stem of the keywords on the query. The keywords are retrieved from the corpus where they are indexed as stems, and then sent back to the Stemmer. The stemmed word or the output is returned back to the user through the user interface.

4.3.3 The Rules

There are about 40 prefixes and 2 suffixes that have been used to design this algorithm and each of them is treated individually. We organized the rules based on the noun classes of the isiXhosa language. First the system checks the length of the input word and passes the word to the next rule on the list for matching the prefixes and suffixes against the ones provided on the list of rules. These rules are presented as follows:

Rule-1

```
if (word. Length () < 4) {
        Print (word + ": The Term is Too Short to be stemmed")
    }
    If(word startsWith a vowel){
        Remove vowel;
            Save words as stem;
        }
```

Rule-2

```
if( word startsWith("asingo-"|| “ayingo-”|| “nga-”|| “asi-”|| “ku-”|| “em-”|| “en-”)){
        Remove Prefix;
        if(name.endsWith("-ana"|| “-kazi” )){
            Remove the suffix}
        }
```

Rule-3

If (word starts with “Um-“) {

Remove the prefix;

If (remaining part ends on ("-ana"|| “-kazi”)) {

Remove the suffix}

}

Rule-4

If (word starts with “Aba- “|| ‘’Abe-‘’ || ‘’Ab-‘’) {

Remove the prefix;

If (remaining part ends on ("-ana"|| “-kazi”)) {

Remove the suffix}

}

Rule-5

If (word starts with “U-“) {

Remove the prefix;

If (remaining part ends on ("-ana"|| “-kazi”)) {

Remove the suffix}

}

Rule-6

If (word starts with “Oo-“) {

Remove the prefix;

If (remaining part ends on ("-ana"|| “-kazi”)) {

Remove the suffix}

}

Rule-7

If (word starts with “Imi-“ || “Im-“) {

Remove the prefix;

If (remaining part ends on ("-ana"|| “-kazi”)) {

Remove the suffix}

}

Rule-8

If (word starts with ‘’Ili-‘’) {

Remove the prefix;

If (remaining part ends on ("-ana"|| “-kazi”)) {

Remove the suffix}

}

Rule-9

If (word starts with ‘’Ama-‘’ || ‘’Ame-‘’) {

Remove the prefix;

If (remaining part ends on ("-ana"|| “-kazi”)) {

Remove the suffix}

}

Rule-10

If (word starts with ‘’Isi-‘’|| ‘’Is-‘’) {

Remove the prefix;

If (remaining part ends on ("-ana"|| “-kazi”)) {

Remove the suffix}

}

Rule-11

If (word starts with ‘’Izi-‘’ || ‘’Iz-‘’) {

Remove the prefix;

If (remaining part ends on ("-ana"|| “-kazi”)) {

Remove the suffix}

}

Rule-12

If (word starts with ‘’In-‘’ || ‘’Im-‘’ || ‘’I-‘’) {

Remove the prefix;

If (remaining part ends on ("-ana"|| “-kazi”)) {

Remove the suffix}

}

Rule-13

If (word starts with ‘’Izin-‘’ || ‘’Izim-‘’ || ‘’Ii-‘’ || ‘’Iin-‘’ ||’’Iim-‘’) {

Remove the prefix;

If (remaining part ends on ("-ana"|| “-kazi”)) {

Remove the suffix}

}

Rule-14

If (word starts with ‘’Ulu-‘’ || ‘’Ulw-‘’ || ‘’Ul-‘’) {

Remove the prefix;

If (remaining part ends on ("-ana"|| “-kazi”)) {

Remove the suffix}

}

Rule-15

If (word starts with ‘’Ubu-‘’ || ‘’Ub-‘’ || ‘’Utyw-‘’ || ‘’Uty-‘’) {

Remove the prefix;

If (remaining part ends on ("-ana"|| “-kazi”)) {

Remove the suffix}

}

Rule-16

If (word starts with ‘’Uku-‘’ || ‘’Uk-‘’ || ‘’Ukw-‘’) {

Remove the prefix;

If (remaining part ends on ("-ana"|| “-kazi”)) {

Remove the suffix}

}

4.4 Conclusion

This Chapter has offered a detailed discussion of the system design, which includes a discussion about the overview of the system architecture. The architecture and the rules that are used by the system are discussed in detail in the Chapter. Other components of the system were also described. The next Chapter discusses the implementation and testing of a Stemmer for the isiXhosa language.

Chapter Five

Implementation

5. Implementation

5.1 Introduction

The objective of this Chapter is to discuss in detail the processes followed in the implementation of the isiXhosa Stemmer. The Chapter explains all the parts of the algorithm that was designed. The Chapter furthermore explains the rules embedded in the algorithm and paying particular attention to special cases where the rule behaves differently from the rest of the other rules. The Chapter concludes by giving a brief description of the system behavior as is implemented.

5.2 Implementation

The system was developed to be used as a tool that can be plugged into an IR system and therefore its interface will not be available to users. It performs its intended function at the back-end of an IR system that uses it. In our case because we wanted to test the Stemmer, we developed an application with the user interface, where users can enter any noun of their choice and it displays its stem on the interface. This application is similar to that of Porter's Stemmer known as the "SNOWBALL" and it is named as "StemXho". The system has a user-interface with a Text field that allows a user to enter a word then the user can click a button to get a stem displayed on another Text box. The user interface of this system is shown in Figure 5.1.

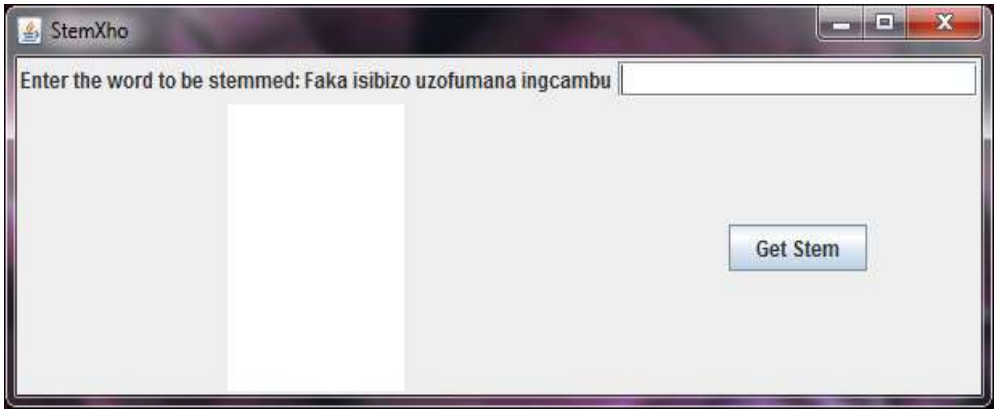

Figure 5.1: StemXho: The IsiXhosa Stemmer

The second form of the user interface of the system is where the system reads a file with different nouns and it checks all the nouns in the file stemming every noun and prints out the stems produced to another file. Actually we used this second form of the system to evaluate and test the accuracy of the Stemmer for the isiXhosa language.

5.3 Modules of the Stemmer

The Stemmer for the isiXhosa language processes each noun in three phases. These include: checking for exceptions, prefix stripping and suffix stripping. We also implemented these phases as separate modules.

5.3.1 Exceptions List Checking

This is the first phase that is followed by the system when it receives a noun as an input. There are words that do not need to be stemmed and the system returns them as they are. As mentioned before, some are included because their length is less than the minimal length of a stemmable noun and therefore they are left untouched.

5.3.2 Prefix Stripping

This is the second step that the algorithm follows when processing a noun. When the word is entered, it is checked if its length is equal or greater than four, then if that is correct it is then this module that checks the noun if it contains the list of compiled prefixes as its prefixes. IsiXhosa forms part of those languages that uses class system, therefore the prefixes used in this step are the noun class prefixes. For research purpose and that we had to include other noun forms, the prefixes for other noun forms were also included. These prefixes do not form part of the noun class prefixes. The algorithm is the longest match and therefore the longest prefix found is removed. The removal of the longest match is achieved by scanning the noun through the list of given prefixes. The prefixes are found in the rules of the stemmer. They are arranged in the order starting from the longest prefix at the top to the shortest at the bottom. Then the remaining part is saved as word and wait for the third phase to begin. Table 5.1 shows the prefix list that was provided during implementation. The prefixes are stored within the rules, because each rule is applied differently in different prefixes. The next example illustrates how the prefixes are stored in the rules of a Stemmer:

If (noun starts with “um”|| “isi”) remove the prefix and store the remaining part of the noun to proceed to the next module.

Table 5.1: Prefix list.

Noun Class	Prefixes	Noun Class	Prefixes
1	Um-	7	Isi-, Is-
2	Aba-, Abe-, Ab-	8	Izi-, Iz-
1 (a)	U-	9	In-, Im-, I-
2 (b)	Oo-	10	Izin-, Izim-, Ii-, Iin-, Iim-
4	Imi-, Im-	11	Ulu-, Ulw-, Ul-
5	Ili-	14	Ubu-, Ub-, Utyw-, Uty-
6	Ama-, Ame-	15	Uku-, Uk-, Ukw-

As can be seen from Table 5.1, that noun class three (3.) is not included in the list of noun classes. The reason for its exclusion is because it is the same as noun class one (1.) and therefore in the prefix stripping phase it doesn't matter whether nouns are of different classes or not but as long as the prefixes are the same the system considers them to be the same. Therefore similar prefixes are removed in the same way regardless of the class they belong to.

5.3.3 Suffix Stripping

Suffix stripping is the third phase of the algorithm. As it is done in the previous phase, this module first checks if the length of the word is greater than four and if so then takes the word and check if it contains a suffix from a list of suffixes that were developed. This step is performed whether a prefix has been found and removed or not. As long as the word has a matching suffix it is removed then the remaining part of the word is saved as the stem. It is also important to note that even for suffix stripping the list of suffixes provided is also that of suffixes that are most occurring or found in nouns. A list of common suffixes that were used in the implementation of the system is provided in Table 5.2.

Table 5.2: Suffix list.

Suffixes
-Ana
-Kazi

Table 5.2 is very small because of the limited number of suffixes that are found in the nouns of the isiXhosa language. From our Corpus we came to realize that there are two most occurring suffixes. These are *Ana* and *Kazi* and the reason for this is that these suffixes are the distinguishing factors of nouns, such as feminine, augment and diminutive which are important in isiXhosa language.

5.4 Putting All Together

The algorithm works by reading all nouns from a text file. For the purpose of this research we created a file called "proj.txt" and include a list of nouns in it, each on a new line. The code segment in Listing 5.1 shows how we read the nouns from this file.

```
File file = new File("proj.txt");
            FileReader fileReader = new FileReader (file);
            BufferedReader bufferedReader = new BufferedReader(fileReader);
            StringBuffer stringBuffer = new StringBuffer();
            String name;
            while ((name = bufferedReader.readLine()) != null) {
```

Listing 5.1: Reading a File

The next step from there would be to look at each noun individually and applying the given rules. In Listing 5.2 we show part of the code segment used to check for the prefix *"aba-"* and the suffixes *"-ana"* and *"-kazi"*. Basically the rules are many and longer and that is the reason why we only present two examples.

```
 if (name.startsWith("aba")){
       name = name.substring(3);
     if (name.endsWith("ana")){
        name = name.substring(0, name.length() - 3);
     }
   if (name.endsWith("kazi")){
      name = name.substring(0, name.length() - 4);
   }
}
```

Listing 5.2: Prefix and Suffix Removal for the Second (2) Noun Class

In the code segment presented in Listing 5.2, we show how the stripping techniques are applied to prefixes and suffixes. In this particular example, the code shows how noun class two affixes: the "*aba-*"prefix and the *"-ana"* and "*-kazi"* suffixes are removed.

5.5 Conclusion

In this Chapter we discussed the implementation stage of the Stemmer. An overview of the stages or components of the Stemmer was also given. Each stage was discussed in detail, starting from checking the exceptions list, prefix stripping to suffix stripping. The following Chapter presents the testing, results, evaluation and discussion.

Chapter Six

Testing, Results and Discussion

6. Testing, Results and Discussion

6.1 Introduction

In the previous Chapter we discussed the system implementation. The current Chapter discusses the system and functionality testing. The results that were collected during the system testing are discussed. The main objective of this Chapter is to analyze and evaluate the results and give meaning to them.

6.2 System Testing

Testing is a very important part of any system development and should be done before a system is deployed. System testing can be conducted in a different number of ways and testing is necessary because it determines whether the system fulfills the requirements or not. In our case the testing was done manually by checking whether or not the system stemmed the nouns correctly. In this testing stage one of the project member, who is a linguist by profession was involved to determine whether the stems generated by the system were correct or not. The testing data set that was used for this purpose was collected randomly from the isiXhosa text books and some from other sources, including research articles in isiXhosa and the isiXhosa dictionary. To be objective in the selection of the test data, every first and third noun after each of the five groups of nouns was taken. This procedure was followed in all the three different isiXhosa text books that were used during the development and testing of the Stemmer. In the dictionary each letter of alphabet had to be represented in the test data. The Stemmer was developed as a java application and testing was performed on the developed application.

6.2.1 Preprocessing

A text editor (notepad in this case) was used for the preparation of the file that contains nouns. A known number of nouns were provided in the file to later check the accuracy of the Stemmer using the stems that were wrongly stemmed or were not stemmed at all.

6.2.2 Test Data

In Table 6.1 we present the sample test data that was randomly collected for testing the Stemmer. This is a small portion taken out of the whole data set. The entire data set used to test the system is attached in Appendix B. It represents nouns that are commonly found in the isiXhosa language but excluding the feminine, diminutive and augmentative representations of a noun. This is

because it forms word set 2, for the repetition of the testing. For the purpose of this presentation the test data that belong to the different noun forms will be presented in different tables. The noun forms representing Feminine, Diminutive and Augmentative are shown in Table 6.2.

Table 6.1: Sample Test Data.

umfo	abafo	into	izinto
umthwa	abathwa	imbabala	iimbabala
umntu	abantu	itafile	iitafile
umoni	aboni	uluvo	Izimvo
umembi	abembi	inkawu	iinkawu
umlilo	imililo	uluthi	izinti
umzi	imizi	ulutho	ulutho
umthi	imithi	ulwazi	ulwazi
		etafileni	yitafile
ilitye	amatye	ulonwabo	ulonwabo
Iliwa	amawa	uphawu	Iimpawu
Ilizwi	amazwi	ubuciko	Ubuciko
ilishwa	amashwa	Kubantu	Ngabantu
ayingobantu		Ebantwini	Asibantu
iqaqa	amaqaqa	utywala	utywala
ithumba	amathumba	utyani	utyani
ifokofoko	amafokofoko	uboya	iimboya
ixaxavhithi	amaxaxavhithi	Bantu	emlanjeni
		Asingobantu	ngamakhwenke
Isiqhamo	iziqhamo	ubusi	ubusi
Isizwe	izizwe	ukutya	ukutya
Isitiya	izitya	ukuphanda	emthonjeni
Isibaluli	izibaluli	ukwaba	entabeni

Table 6.2: Sample Test Data Showing the Feminine, Diminutive and Augmentative.

FEMININE	DIMINUTIVE	AUGMENTATIVE
inkondekazi	umfana	umfokazi
inkosikazi	indodana	umthikazi
ixhegokazi	incwadana	intabakazi
iqabakazi	idolophana	umlambokazi
umkhuluwakazi	umhlatyana	isizwekazi
umninawakazi	idwalana	uphawukazi
igqirhakazi	iqobokazana	incebakazi

6.2.3 Functionality Testing

Functionality testing is a method used for checking the system for all the functionalities that is required and specified in the functionality requirements gathering (McGregor & Sykes, 2001). It is in this stage that we used the test data presented in Appendix B to check if the system stems the nouns correctly.

6.3 Evaluation

Stemmers can be evaluated in many different methods and these include compression performance where the Stemmers are compared on how much they compress the index files, retrieval effectiveness and correctness. Any of these criteria can be judged mainly using the two types of errors that can be found in any Stemmer, which are over-stemming and under-stemming. Under-stemming occurs when a prefix or a suffix that is supposed to be removed from a noun but the system fails to remove it. Over-stemming, on the other hand refers to the case where prefixes or suffixes that were not supposed to be removed are removed by the system. As mentioned before, the stemmed words are evaluated by Dr. Zoliswa Mali (one of the co-supervisors of this dissertation). The results were obtained from the testing data set that is presented in Appendix B. The test data set comprised of randomly selected nouns. These nouns were all read from the same file where each noun was stemmed according to the rules specified

in the design of the Stemmer. After reading each noun the system then passes it through all the stages of checking whether it is exception, or contains prefixes or suffixes to be removed. After those three stages have been performed the system presents the corresponding stem. The output of the system (stems) is presented in Appendix A as results, where each noun is shown next to its corresponding stem as labeled by the system and the incorrect stems are bold and highlighted in red.

6.4 Results

The table in Appendix A shows results that were obtained from the test data. Some nouns were incorrectly stemmed due to over-stemming and under-stemming. The combined results of the evaluation are shown as a pie chart in Figure 6.1.

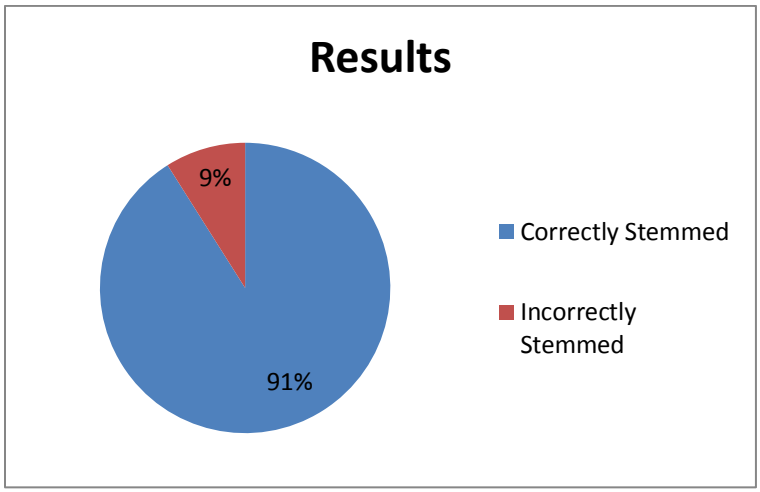

Figure 6.1: Test Results

Figure 6.2 illustrates the distribution of errors. There are two types of errors as mentioned in the evaluation and they are under stemming and over stemming. For our algorithm there were no under stemming errors. The sets that were used for testing are summarized in Table 6.3

Table 6.3: Results

Number of words	Correctly Stemmed	Incorrectly Stemmed
Word set one: 404	376	28
Word set two: 120	101	19

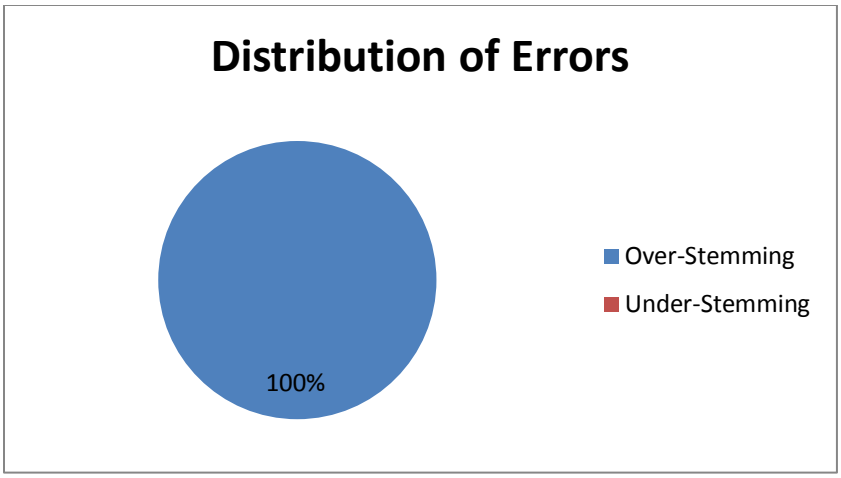

Figure 6.2: Distribution of Errors

We can say that the algorithm is working well with a good precision as the number of errors is only 9 %. As can be seen from the Figure, the Stemmer is aggressive because the only form of errors found is over-stemming.

6.5 Discussion

The Stemmer is found to be 91% accurate; this is based on the sample data that was selected to do the testing. Even though it may differ from language to language, it is impossible to achieve 100% accuracy. The stripping techniques are applied in the same way for all the nouns. Some of the nouns do not have suffixes; they only have a prefix to be removed. Nouns such as; *aboni* "sinners", *umntu* "person", *imililo* "fires" etc, only have prefixes to be removed. It is found that in some nouns the algorithm result in over-stemming. It was mentioned before that this is a rule based stemming algorithm and therefore for this reason it is difficult to create rules that cater for all the possible nouns in the language. If one tries to accommodate all the different variations of nouns, the accuracy of the Stemmer may deteriorate. The removal of the suffix ***–ana***, appears to be the greatest cause of over-stemming and this requires further investigation. The suffix ***–ana*** is used to denote the diminutive in the isiXhosa language. It was mentioned previously that in isiXhosa all the stems end in a vowel, but in the case of ***–ana*** the vowel ***-a-*** is removed with the whole suffix and leaving a noun that is over-stemmed. This suffix requires a special rule to be designed for it, a rule that leave the **-a-** before –**na** so that the stem end in a vowel. However, we tried to include a rule but could not work for all the nouns that have this suffix, and the more we

try to be accurate with this one suffix, the performance of the Stemmer somehow deteriorated. So, further investigation needs to be done on this part.

Because the main purpose of stemming in IR systems is to bring together the different forms of a word into a single term that represent all the variations of those words. We can develop a Stemmer that creates stems without a vowel at the end as long as Stemmer brings all the variations of a word into a single representation. However, as mentioned before, for this project we followed the rules of the isiXhosa language and decided to use stems with the final vowel.

Verbal and/or nominal roots that are mono-syllabic may be troublesome. Let us look at the following examples that we may get as stems if we decide not to use the proper stems that ends in a vowel: ***-f- ; -m-; -hl-; -v-,*** this is very difficult to say what it stands for and therefore for this reason stemming for this project was performed differently from other languages such as English where a stem may not contain a vowel at the end. The isiXhosa language has a rich set of prefixes and fortunately the prefix stripping part can be considered a success since there are very few errors that arose because of incorrect removal of a prefix. Therefore the algorithm seems to be problematic in suffix removal but one suffix in particular, ***-ana.*** In general, we can say that the algorithm serves its purpose of bringing different forms of a word into a single representation. It is evident from the results that if a word is given in plural form and also in its singular form, the results after stemming are the same. Therefore from these results we conclude that the Stemmer was successfully implemented and it can be used in IR systems to help with query broadening of isiXhosa documents.

6.6 Problems Related To Noun Classes

In Chapter Three, it is mentioned that a noun can occur in different forms, such forms are; nominal, vocative, locative, predicative/copulative and negative predicative. These forms of nouns have special prefixes that are not part of the noun class prefixes and therefore they had to be included in the rules of the algorithm except the nominal nouns. In some instances where these nouns are used it is difficult to convert their stems to represent a common form. Let us look at the example; *umlambo* “river”, this is a nominal form of a noun and now its locative form is *emlanjeni* “at the river”. Their stems are as follows:

umlambo = -lambo

emlanjeni= -lanjeni

For retrieval purposes it might be difficult to conflate these terms to a single representation because their stems are different. Our suggestion is that each noun form can have its own stems without being related to other form, even though the meaning may be the same. In the case of a predicative/copulative form we may have *itafile* "table" as its nominal form and *yitafile* "is a table" as its predicative form. When these two terms are stemmed we get:

itafile =-tafile

yitafile= -tafile

These terms are conflated into a single representation and therefore they are both retrieved and our algorithm shows a high accuracy in this noun form (predicative).

Another noun form that we look at is a negative predicative, as the name says; it is a negative of the predicative. This form also has its own types of prefixes that are not found in the noun class prefixes. Let us look at the example and see how our algorithm handled this noun form: *ngamakhwenkwe* "they are boys" and the negative is; *asingomakhwenkwe* "they are no boys". When these two terms are entered in our Stemmer they are stemmed as:

Ngamakhwenkwe = -khwenkwe

Asingomakhwenkwe= -khwenkwe

As is the case with the predicative form of a noun the Stemmer was able to conflate these terms into a single representation and therefore increasing the accuracy of a Stemmer. These noun forms are very important in any language because in IR systems users enter queries using natural language and therefore the Stemmer needs to take all the noun forms and conflate as many terms as possible.

The last noun form we looked at is a vocative form and this noun form is formed by dropping the augment from the nominal form of a noun. This is the most commonly occurring form of a noun following the nominal form. The following example shows how our Stemmer acted in this case: *mntu* "person" in its singular and *bantu* "people" the plural form. When these two terms are conflated they give:

mntu = -ntu

bantu = -ntu

The vocative form of a noun also showed a high accuracy in conflating different terms to a single term. The Stemmer also showed over stemming in this form where there is a prefix **–ana**. In

cases such as; *ncwadi* “book” and *ncwadana* “small book” which denotes a diminutive. The stems that were produced are:

ncwadi = -cwadi

ncwadana= -cwad-

There are those nouns that were not stemmed correctly, due to the differences in the noun forms. We can see the performance of the Stemmer improving in the future, if we work on what we have already discovered.

6.6 Conclussion

In this Chapter we presented testing and the results and also the evaluation of the results. The Chapter concluded by the detailed discussion of the results, giving possible explation on why we got the kind of results that we ended up with. The next Chapter will talk about the achievements of the objectives, research summary and the future work.

Chapter Seven

Conclusion

7. Conclusion

7.1 Introduction

In the previous Chapter, we discussed the results, evaluation and discussion of the results; in the present Chapter we present summary of the whole work. We also present the achievement of the objectives that we discussed in Chapter One of the dissertation. The Chapter concludes by discussing the future work.

7.2 Research Summary

IsiXhosa is an agglutinative language, meaning affixes glue to each other to form different variations of words. The main word formation in isiXhosa is through affixation. In other words, IsiXhosa uses prefixes, suffixes and roots/stems. Information Retrieval (IR) systems mostly use nouns as documents representations and these keywords should be stemmed before they are used as document representations. Users also submit their queries by assuming the keywords that may exist in documents that they are looking for. So similarly users' queries also need to be stemmed. This means that a Stemmer is the major part of any IR system. We viewed different stemming algorithms developed for other languages. Most of these algorithms make use of the well-known Porter's stemming algorithm which was developed for the English language. However because of the morphological complexity and difference in features of the isiXhosa language, we could not use the Porter's algorithm in its totality. Nevertheless some of the rules and techniques that are applicable to isiXhosa language are taken from Porter. From our investigation of the linguistics of isiXhosa, we came to realize that affixes are the most important aspects of word formation and we designed and developed affix removal technique for the isiXhosa nouns.

Due to the fact that isiXhosa word formation is mainly through affixation, we therefore decided that prefix and suffix removal was enough for the isiXhosa language, because isiXhosa uses a noun class system. In a noun class system the prefixes are grouped according to their classes. The grouping of the noun classes made it easy to develop the rule-based affix stripping algorithm for the isiXhosa language. The rules are discussed in Chapter Four of this dissertation and were processed in the following order; exception checking prefixes and suffixes removal. The system was tested using test data that was randomly collected. We also considered all the different forms of the isiXhosa nouns when collecting the test data. The results of the stemming process showed some errors particularly in the locative forms of nouns. We also came to understand that a

special rule needs to be designed specifically for the suffix –**ana** as this suffix is the major cause of over stemming. From the evaluation of the results we can conclude that we have achieved our goal of developing a Stemmer that considers all the different forms of nouns. The results show 91 % accuracy. The distribution of errors is 9% over-stemming and 0% under-stemming. The result is promising and this means using this Stemmer in IR systems could help conflate isiXhosa keywords and also help increase query broadening.

7.3 Achievements of the Objectives

The main objective of the study was to develop and implement the noun Stemmer for the isiXhosa language that will help conflate isiXhosa nouns for information retrieval purposes. This objective was achieved by consulting a lot of literature on the already developed Stemmers and by studying the morphology of the isiXhosa language. The dissertation had different sub-objectives and the following paragraphs show how the sub-objectives of the dissertation are achieved.

Objective 1: Investigate how the stemming algorithms are developed and how they work.

This sub-objective was achieved by conducting an extensive literature survey that is presented in Chapter two, where we studied stemming algorithms for other languages and learned the rules that represent them. From the survey we noticed that Stemmers are language dependent and therefore it is important for one to study the morphology of the language of interest.

Objective 2: Investigate the rules of the isiXhosa as a written language and the noun formation in the language.

The second sub-objective was to study the language, i.e., how it is written and how the noun formation is carried out. In general, this includes studying the internal morphology of the language. This sub-objective was achieved by carrying out literature survey on the language and our analysis is presented in Chapter three.

Objective 3: Finding a suitable stemming algorithm that can be used as a reference in building the isiXhosa noun Stemmer.

The third sub-objective was to study the already developed stemming algorithms and to check if any of those algorithms can be used as a reference in developing a noun Stemmer for the

isiXhosa language. It was found out that most researchers used the well-known Porters algorithm when developing stemming algorithms for other languages other than English. The iterative technique that was used by Porter could not be applied for the isiXhosa language because the prefixes are standard in isiXhosa. But again the simplicity and directness of Porters algorithm, as discussed in Chapter two, was found to be good and we followed that aspect when trying to develop the Stemmer.

Objective 4: Developing a noun Stemmer for the isiXhosa language.

The fourth sub-objective was achieved by collecting all the relevant information that was obtained in the above sub-objectives. All the gathered information was put together so as to develop the isiXhosa Stemmer. The design of this Stemmer is presented in Chapter four and its implementation is presented in Chapter five of this dissertation.

Objective 5: Test and Evaluate the Stemmer based on the available test data that was collected from the isiXhosa text books.

The fifth sub-objective was achieved by collecting a random data sample from three different isiXhosa text books and from the isiXhosa dictionary (*Isichazi-Magama SesiXhosa)*. The collected data was entered into a file and then read by the Stemmer to process them using stemming rules. The results were printed out on another file, which was given to a linguist to check the number of the correct and incorrect stems that were generated by the Stemmer. The result of this activity is presented in Chapter five and Chapter six.

7.4 Future Work

The development of a noun Stemmer for the isiXhosa language was successfully carried out and the results were also very promising. Similar to any other system, the Stemmer has some drawbacks and needs further improvements regarding the improvement of the algorithm and the efficiency of the stemming process. The main focus of the study was on noun stemming. Therefore, in the future the Stemmer can be extended to include also verbs as they constitute a very important role in a language. The Stemmer was designed and tested in a small text size and it has never been put on an IR system like Google or any other serach engine. Therefore in order to determine its accuracy it is wise in the future to use a larger test collection where all the possible nouns can be found. Also in the larger text size different characteristics of the language

can be found. The operation of the affix stripping depends on the rules that are supplied in the algorithm and therefore the algorithm does not differentiate between word forms or types of the words. The algorithm only takes a noun and performs stripping based on the applying rules that are suitable for that particular noun. If maybe the Stemmer could be incorporated with a Part-Of-Speech-Tagger that tells the Stemmer the type or form of the word to increase its accuracy.

More research in this field can be helpful to develop more application tools of the isiXhosa language. Application tools such as spell checker, lemmatizer and thesaurus are some applications of a Stemmer. These applications need stemming in order to perform well and therefore this research can be a starting point of other language related researches for isiXhosa language. Not much language related research is done for isiXhosa language and we believe if more work can be done in this field the country can benefit a lot and the teaching and learning process would be easier as students and educators will be supported by IR systems to get relevant documents. So we encourage the integration of the Stemmer in IR systems like Google to assist the teaching and learning process.

8. References

Abedo, M. K. (2012). *Designing a stemming algorithm for the Slit'e language*. Unpublished Masters Dissertation, Addis Ababa, Addis Ababa University.

Adamson, G. W., & Boreham, J. (1974). The use of an association measure based on character structure to identify semantically related pairs of words and document titles. *Information Storage and Retrieval*, *10*(7-8), 253–260.

Ahmad, F., Yusoff, M., & Sembok, T. M. T. (1996). Experiments with a stemming algorithm for Malay words. *Journal of the American Society for Information Science*, *47*(12), 909–918.

Ahmad, S. (2007). Tutorial on Natural Language Processing. *Artificial Intelligence*, 810(161).

Alexandersson, M., Runeson, P. & Nyholm, O. (2007). Detection of duplicate defect reports using natural language processing. In *Software Engineering ICSE 2007, 29th International Conference*. IEEE. 499-510.

Al-Shalabi, R., Kannan, G., Hilat, I., Ababneh, A., & Al-Zubi, A. (2005). Experiments with the successor variety algorithm using the cutoff and entropy methods. *Information Technology Journal*, *4*(1), 55-62.

Ali, N. H., & Ibrahim, N. S. (2012). Porter Stemming Algorithm for Semantic Checking. *In International Conference on Communications and Information Technology*.

Bennie, W.G. (1931). A Practical Orthography for Xhosa. Union Government Advisory Committee on Bantu Studies and Research, Johannesburg.

Brown, A. (2013). International Phonetic Alphabet. *In The Encyclopedia of Applied Linguistics*. Wiley Online Library.

Cherpas, C. (1992). Natural language processing, pragmatics, and verbal behavior. *The Analysis of verbal behavior*. In Marcel Decker (ed), Encyclopedia of Library and Information Science. *10*, 135–47.

Chris, D. P. (1990). Another Stemmer, In *ACM SIGIR Forum.* 24(3), 56-61.

Craig, C. G. (Ed.). (1986). *Noun classes and categorization: proceedings of a symposium on categorization and noun classification.* John Benjamins Publishing Company, *Eugene, Oregon,* (vol. 7).

Dawson, J. (1974). Suffix removal and word conflation, *ALLC bulletin. 2*(3), 33-46.

Du Plessis, J.A., Visser, M. (1992). *Xhosa Syntax*, Via Afrika Limited.

Fotinea, S. F., Tambouratzis, G. D., & Carayannis, G. V. (2001). *Constructing a Segment Database for Greek Time Domain Speech Synthesis*. In *INTERSPEECH* (pp. 2075-2078).

Geeraerts, D. (Ed.). (2006). *Cognitive linguistics: basic readings*. Walter de Gruyter.

Goldsmith, J. A., (2001). Unsupervised learning of the morphology of a natural language, *Computational Linguistics*, 27(2), 153-198.

Hendrix, G. G., & Lewis, W. H. (1981). Transportable natural-language interfaces to databases. In *Proceedings of the 19th annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 159-165.

International Phonetic Association (Ed.). (1999). *Handbook of the International Phonetic Association: A guide to the use of the International Phonetic Alphabet*. Cambridge University Press.

Islam, Z., Uddin, N., & Khan, M. (2007). A Light Weight Stemmer for Bengali and Its Use in Spelling Checker. Center for Research on Bangla Language Processing (CRBLP), BRAC University.

Jokweni, M. W. (1997). Identifying, descriptive and associative copulatives in Xhosa structure and function. *South African Journal of African Languages*. *17*(3), 110-113.

Liddy, E.D. (2001). Natural Language Processing. In Encyclopedia of Library and Information Science, 2nd ed. NY. Marcel Decker, Inc.

Lovins, J. B. (1968). Development of a stemming algorithm. MIT Information Processing Group. Electronic Systems Laboratory, 22-31.

Majumder, P., Mitra, M., Parui, S. K., Kole, G., Mitra, P., & Datta, K. (2007). "YASS: Yet another suffix stripper", ACM *transactions on information systems (TOIS)*, *25*(4).

McGregor, J. D., & Sykes, D. A. (2001). *A practical guide to testing object-oriented software*. Addison-Wesley Professional. United States of America.

Melucci, M., & Orio, N. (2003). A novel method for Stemmer generation based on hidden Markov models. ACM, In *Proceedings of the twelfth international conference on Information and knowledge management*. 131-138.

Mtuze, P. T., Tshabe, S.L., Putu, B.D., Mini, B.M. & Mkonto, N.V. (1987). *IsiXhosa Sezikhuthali*. De Jager-Haum

Ntais, G. (2006). Development of a Stemmer for the Greek Language. Masters dissertation, Sweden, Stockholm University / Royal Institute of Technology.

Othman, A. (1993). Pengakar perkataan melayu untuk sistem capaian dokumen. Unpublished Masters Dissertation. National University of Malaysia.

Paice, C. D. (1996). Method for evaluation of stemming algorithms based on error counting. *Journal of the American Society for Information Science*, *47*(8), 632-649.

Paice, C. D., & Husk, G. D. (1987). Towards the automatic recognition of anaphoric features in English text: the impersonal pronoun "it". *Computer Speech & Language*, *2*(2), 109-132.

Pascoe, M., & Smouse, M. (2012). Masithethe: speech and language development and difficulties in isiXhosa. *South African medical journal*. *102*(6), 469–71.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program: electronic library and information systems*. *14*(3), 130-137.

Satyo, S. (1988). *Igrama noncwadi lwesixhosa ibanga 10*. Via Afrika Limited.

Saul, Z.W. (1998). The Orthographical Development of IsiXhosa Since 1824. Doctoral dissertation. Alice, University of Fort Hare.

Sharma, D. (2012). Stemming algorithms: A comparative study and their analysis. *International Journal of Applied Information Systems*. 4(3), 7-12.

Smirnov, I. (2008). Overview of stemming algorithms. *Mechanical Translation*. DePaul University.

Statistics South Africa 2011 (2012) Available at: http://www.statssa.gov.za/census2011/default.asp

Tshabe,S.L., Guzana,Z., & Nokele, A.B.B. (2008). *Isichazi-Magama sesiXhosa*. Nutrend Publishers.

Tomita, M., & Carbonell, J. G. (1986). Another stride towards knowledge-based machine translation. In *Proceedings of the 11th conference on Computational linguistics*. Association for Computational Linguistics. Stroudsburg, PA, USA, 633-638.

Willett, P. (2006). The Porter stemming algorithm: then and now. *Program: electronic library and information systems*. *40*(3), 219–223.

9. Appendix A-Results

Nouns	Corresponding Stems
umfo	fo
umthwa	thwa
umntu	ntu
umoni	**oni**
umembi	**embi**
abafo	fo
abathwa	thwa
abantu	ntu
aboni	**oni**
abembi	mbi
umlilo	lilo
umzi	zi
umthi	thi
imililo	lilo
imizi	zi
imithi	thi
ilitye	tye
iliwa	wa
ilizwi	zwi
ilishwa	shwa
amatye	tye
amawa	wa
amazwi	zwi
amashwa	shwa
iqaqa	qaqa
ithumba	thumba
ifokofoko	fokofoko
ixaxavithi	xaxavithi
amaqaqa	qaqa
amathumba	thumba
amafokofoko	fokofoko
amaxaxaviti	xaxavithi
isiqhamo	qhamo
isizwe	zwe
isitiya	tiya
isibaluli	baluli
isazi	**azi**
iziqhamo	qhamo
izizwe	zwe
izitiya	tiya
izibaluli	baluli

izazi	**azi**
into	to
imbabala	**babala**
itafile	tafile
inkawu	**kawu**
uluvo	vo
izinto	nto
iimbabala	mbabala
iitafile	tafile
izimvo	mvo
iinkawu	nkawu
uluthi	thi
ulutho	tho
ulwazi	**azi**
ulonwabo	**onwabo**
uphawu	phawu
ubuciko	ciko
utywala	**ala**
utyani	**ani**
uboya	**oya**
ubusi	**si**
iimboya	mboya
ukutya	tya
ukuphanda	phanda
ukwaba	**aba**
ukona	**ona**
inkondekazi	konde
inkosikazi	kosi
ixhegokazi	xhego
iqabakazi	qaba
umkhuluwakazi	khuluwa
umninawakazi	ninawa
igqirhakazi	qqirha
umfana	**f**
indodana	**dod**
incwadana	**cwad**
idolophana	**doloph**
umhlatyana	**hlaty**
idwalana	**dwal**
iqobokazana	**qobokaz**
umfokazi	fo
umthikazi	thi
intabakazi	taba

umlambokazi	lambo
isizwekazi	zwe
uphawukazi	phawu
incebakazi	ceba
asingobantu	bantu
ayingobantu	bantu
ngabantu	bantu
emlanjeni	lanjeni
entabeni	tabeni
emthonjeni	thonjeni
asibantu	bantu
nja	nja
fa	fa
umdlalo	dlalo
uthando	thando
intando	tando
isono	ono
umbongo	bongo
umzamo	zamo
umculo	culo
itshivela	tshivela
ithemba	themba
izola	**ola**
ihlaba	hlaba
igeza	geza
umthunywa	thunywa
isithunywa	thunywa
umbotshwa	botshwa
isibotshwa	botshwa
umfunzwa	funzwa
ithokazi	tho
injakazi	**ja**
indlovukazi	dlovu
uthulikazi	thuli
isithilikazi	thili
isithili	thili
ilifana	**f**
indulana	**dul**
inkobonkobo	kobonkobo
inkobonkobana	kobonkob
amanzana	**nz**

inembana	**emb**
umhluzana	**hluz**
ubukhulu	khulu
ubufutshane	futshane
ubude	de
ubumanzi	manzi
ububanzi	banzi
ubungqoqo	ngqoqo
unqaku	nqaku
uthwahla	thwahla
ugovo	govo
uthwayithwayi	thwayithwayi
isiqabu	qabu
umbane	bane
umphezulu	phezulu
umphantsi	phantsi
umphandle	phandle
umphakati	phakati
amahayihayi	hayihayi
amayeye	yeye
sonininanini	sonininanini
usonininanini	sonininanini
somandla	somandla
usomandla	somandla
sotheko	sotheko
usotheko	sotheko
sosuthu	sosuthu
usosuthu	sosuthu
somanyala	somanyala
usomanyala	somanyala
iindima	dima
umkhwetha	khwetha
umzala	zala
umzali	zali
afrika	afrika
umafrika	afrika
ndebele	ndebele
umndebele	ndebele
xhosa	xhosa
umxhosa	xhosa
umphuphi	phuphi

umphaki	phaki
umpheki	pheki
unyoko	nyoko
gugulethu	gugulethu
ugugulethu	gugulethu
nomasele	nomasele
unomasele	nomasele
abakhwetha	khwetha
abazala	zala
abazali	zali
oonyoko	nyoko
oogugulethu	gugulethu
umthuma	thuma
ithimla	thimla
inqilo	qilo
amaxhosa	xhosa
amaciko	ciko
isicithi	cithi
izicithi	cithi
isabelo	abelo
izabelo	abelo
isikhutali	khutali
izikhuthali	khuthali
isinayimesi	nayimesi
izinayimesi	ayimesi
intso	tso
izintso	tso
ihoko	hoko
ubukhwenkwe	khwenkwe
imibhalo	bhalo
iilwimi	lwimi
ukwandlala	**andlala**
iindywala	dywala
umculi	culi
isabatha	abatha
isabelo	abelo
umabeli	**abeli**
umabelwa	abelwa
isabhalala	abhalala
ulwabhici	**abhici**
isabhokhwe	abhokhwe

isabhongo	abhongo
isabhunge	abhunge
isabhungqu	abhungqu
izabhungqu	abhungqu
umabi	**abi**
isacholo	acholo
isaci	**aci**
isadunge	adunge
isadyenge	adyenge
isadyudyu	adyudyu
isanda	**anda**
isandanda	andanda
isandulela	andulela
umanyiselelwa	**anyiselelwa**
ulwapesi	**apesi**
ulwatshaka	**atshaka**
ubabalo	babalo
ubafazini	**afazini**
ibakwana	**bakw**
umbandavu	bandavu
umbandela	bandela
ibandla	bandla
ubandlululo	andlululo
ibanga	banga
ibanjwa	banjwa
umbanjwa	banjwa
ibatha	batha
umbathalala	bathalala
ibekelo	bekelo
ibele	bele
umbeko	beko
umbele	bele
ububele	bele
isibeleko	beleko
ubengezelo	engezelo
umbengo	bengo
ubende	ende
ubengu	bengu
umbethe	bethe
isibetho	betho
isibhakabhaka	bhakabhaka

umbhako	bhako
umbhali	bhali
isibhanxa	bhanxa
isibhaxu	bhaxu
ubhengxeshe	hengxeshe
ibhinqa	bhinqa
ibhobhile	bhobhile
umbhobho	bhobho
umbhiyozo	bhiyozo
ubhobhoyi	hobhoyi
umbhodamo	bhodamo
ubhontsi	hontsi
icabanga	cabanga
ucaca	caca
isicaka	caka
icala	cala
icamagu	camagu
icandelo	candelo
ucanzibe	canzibe
icawa	cawa
icebiso	cebiso
iceba	ceba
icekwa	cekwa
umcephe	cephe
icephe	cephe
isicengo	cengo
iceya	ceya
izicheko	cheko
ichele	chele
isichenge	chenge
isichenene	chenene
ichitywa	chitywa
amacholi	choli
umcholwa	cholwa
isichopho	chopho
ubuchopo	chopo
isichotho	chotho
uchuku	chuku
ichule	chule
ubuchule	chule
uchulumanco	chulumanco

isichumiso	chumiso
umchwebelele	chwebelele
ubuchwepheshe	chwepheshe
ucikicane	cikicane
umcinga	cinga
ucingo	cingo
amacirha	cirha
isicithi	cithi
icongwane	congwane
umcondo	condo
umda	da
udaba	daba
isidabane	dabane
udakada	dakada
isidala	dala
umdala	dala
isidalwa	dalwa
isidamba	damba
isidanga	danga
idangatye	dangatye
idelakufa	delakufa
isidenge	denge
isidikimfa	dikimfa
isidima	dima
udlalani	dlalani
umdlanga	dlanga
isidlangalala	dlangalala
idlavu	dlavu
idlelo	dlelo
umdlezane	dlezane
udlezinye	dlezinye
idliso	dliso
ukudliwa	dliwa
isidlo	dlo
isidlokolo	dlokolo
idlolo	dlolo
udlomdlayo	dlomdlayo
umdlungu	dlungu
udlwabevu	dlwabevu
isidlwengu	dlwengu
idobo	dobo

ubudoda	doda
isidodo	dodo
umdolomba	dolomba
isidudla	dudla
idyudyu	dyudyu
ifanankosi	fanankosi
umfanekiso	fanekiso
umfaneleko	faneleko
umfazana	**faz**
isifede	fede
ufefe	fefe
isifingo	fingo
umfinya	finya
isifinyezo	finyezo
umfuxane	fuxane
umfuyi	fuyi
igaba	gaba
umgada	gada
igalelo	galelo
umgama	gama
umgaqo	gaqo
igazi	gazi
ugcado	gcado
igcisa	gcisa
ukugxwala	gxwala
ihempe	hempe
ubuhilihili	hilihili
umhla	hla
ihlaba	hlaba
umhlabangubo	hlabangubo
umhlabeli	hlabeli
umhlakulo	hlakulo
isihlanganisi	hlanganisi
isihlangu	hlangu
uhlaselo	hlaselo
isihlava	hlava
umhlekazi	hle
umhlekisi	hlekisi
umhleli	hleli
ubuhlobo	hlobo
isihlomelo	hlomelo

ukwindla	ndla
ijaja	jaja
ijaji	jaji
ijengxeba	jengxeba
ijikazi	ji
ujingi	jingi
ujobela	jobela
ikamva	kamva
ikhaba	khaba
isikhaka	khaka
isikhakhamela	khakhamela
ikhakhasholo	khakhasholo
isikhala	khala
isikhalazo	khalazo
isikhali	khali
ikhangala	khangala
umkhandi	khandi
ikhatshu	khatshu
umkhonzi	khonzi
ilahleko	lahleko
ilalela	lalela
ilangatya	langatya
ilaphu	laphu
umlawuli	lawuli
ubulembu	lembu
isimahla	mahla
umahlalela	ahlalela
isimamva	mamva
isimelabizo	melabizo
imfeketho	feketho
isinciphiso	nciphiso
incindi	cindi
umncethezi	ncethezi
inceku	ceku
indawo	dawo
indebe	debe
indlezana	dlez
indlamanzi	dlamanzi
indlazi	dlazi
indwadunge	dwadunge
indwalutho	dwalutho

ingcathu	gcathu
ingxangxa	gxangxa
ingxangxasi	gxangxasi
ingxangxosi	gxangxosi
isomelezo	omelezo
umondli	ondli
umonakalo	**noakalo**
umongi	**ongi**
umongo	**ongo**
isonka	onka
upapasho	papasho
umpapasho	papasho
ipasika	pasika
ipasile	pasile
isipeliti	peliti
ipatyutyu	patyutyu
uphahla	phahla
isiphako	phako
umphako	phako
umphanda	phanda
iphanyazo	phanyazo
isiphango	phango
iqabaza	qabaza
uqabisisu	qabisisu
iqadi	qadi
umqala	qala
iqaqoba	qaqoba
umqaqoba	qaqoba
uqavashe	qavashe
iqela	qela
iqegu	qegu
umqhagi	qhagi
isiqhazolo	qhazolo
umqhele	qhele
irhabasa	rhabasa
irhafu	rhafu
irhamncwa	rhamncwa
irharha	rharha
urhexe	rhexe
irhengqe	rhengqe
irhewu	rhewu

umrhumo	rhumo
umrhubhe	rhubhe
isabile	abile
isandle	andle
umsantsa	santsa
isamani	amani
iselwa	**elwa**
isishiqi	shiqi
umshologu	shologu
isishuba	shuba
ishwangusha	hwangusha
ubusika	sika
isisila	sila
usinga	singa
umtha	tha
isitha	tha
umthabalala	thabalala
isithako	thako
ithambeka	thambeka
ithambo	thambo
imithombo	thombo
isithombo	thombo
isithonga	thonga
ithongo	thongo
imvaba	vaba
umvambo	vambo
uvalo	valo
umvandedwa	vandedwa
ivamna	vamna
imvana	**v**
imvelo	velo
uvelwano	velwano
iliwa	wa
iwaka	waka
umwethu	wethu
isiwiliwili	wiliwili
isixando	xando
uxam	xam
ixandeka	xandeka
ixethuka	xethuka
umyalelo	yalelo

umyalezo	yalezo
iyelenqe	yelenqe
uyihlo	yihlo
uyihlokazi	yihlo
uyisemkhulu	yisemkhulu
uyisemncinci	yisemncinci
uyisekazi	yise
isiza	za
umzabalazo	zabalazo
umzali	zali
umzalikazi	zali
umzalwana	zalw
isizamva	zamva
isizananina	zananina
isizekabani	zekabani
isiziba	ziba
izibazana	**baz**
izibuko	buko
isizungu	zungu
emthonjeni	thonjeni
entabeni	tabeni
asingomakhwenkwe	makhwenkwe
emlanjeni	lanjeni
ngamakhwenke	makhwenke
bantu	bantu
asingobantu	bantu
ngabantu	bantu
asibantu	bantu
kubantu	bantu
ebantwini	ntwini

10. Appendix B- Test Data

umfo
umthwa
umntu
umoni
umembi
abafo
abathwa
abantu
aboni
abembi
umlilo
umzi
umthi
imililo
imizi
imithi
ilitye
iliwa
ilizwi
ilishwa
amatye
amawa
amazwi
amashwa
iqaqa
ithumba
ifokofoko
ixaxavithi
amaqaqa
amathumba
amafokofoko
amaxaxavhithi
isiqhamo
isizwe
isitiya
isibaluli
isazi
iziqhamo
izizwe
izitiya
izibaluli
izazi
into

imbabala
itafile
inkawu
uluvo
izinto
iimbabala
iitafile
izimvo
iinkawu
uluthi
ulutho
ulwazi
ulonwabo
uphawu
ubuciko
utywala
utyani
uboya
ubusi
iimboya
ukutya
ukuphanda
ukwaba
ukona
inkondekazi
inkosikazi
ixhegokazi
iqabakazi
umkhuluwakazi
umninawakazi
igqirhakazi
umfana
indodana
incwadana
idolophana
umhlatyana
idwalana
iqobokazana
umfokazi
umthikazi
intabakazi
umlambokazi
isizwekazi

uphawukazi
incebakazi
asingobantu
ayingobantu
ngabantu
emlanjeni
entabeni
emthonjeni
asibantu
nja
fa
umdlalo
uthando
intando
isono
umbongo
umzamo
umculo
itshivela
ithemba
izola
ihlaba
igeza
umthunywa
isithunywa
umbotshwa
isibotshwa
umfunzwa
ithokazi
injakazi
indlovukazi
uthulikazi
isithilikazi
isithili
ilifana
indulana
inkobonkobo
inkobonkobana
amanzana
inembana
umhluzana
ubukhulu
ubufutshane

ubude
ubumanzi
ububanzi
ubungqoqo
unqaku
uthwahla
ugovo
uthwayithwayi
isiqabu
umbane
umphezulu
umphantsi
umphandle
umphakati
amahayihayi
amayeye
sonininanini
usonininanini
somandla
usomandla
sotheko
usotheko
sosuthu
usosuthu
somanyala
usomanyala
iindima
umkhwetha
umzala
umzali
afrika
umafrika
ndebele
umndebele
xhosa
umxhosa
umphuphi
umphaki
umpheki
unyoko
gugulethu
ugugulethu
nomasele
unomasele
abakhwetha
abazala

abazali
oonyoko
oogugulethu
umthuma
ithimla
inqilo
amaxhosa
amaciko
isicithi
izicithi
isabelo
izabelo
isikhutali
izikhuthali
isinayimesi
izinayimesi
intso
izintso
ihoko
ubukhwenkwe
imibhalo
iilwimi
ukwandlala
iindywala
umculi
isabatha
isabelo
umabeli
umabelwa
isabhalala
ulwabhici
isabhokhwe
isabhongo
isabhunge
isabhunqu
izabhungqu
umabi
isacholo
isaci
isadunge
isadyenge
isadyudyu
isanda
isandanda
isandulela
umanyiselelwa

ulwapesi
ulwatshaka
ubabalo
ubafazini
ibakwana
umbandavu
umbandela
ibandla
ubandlululo
ibanga
ibanjwa
umbanjwa
ibatha
umbathalala
ibekelo
ibele
umbeko
umbele
ububele
isibeleko
ubengezelo
umbengo
ubende
ubengu
umbethe
isibetho
isibhakabhaka
umbhako
umbhali
isibhanxa
isibhaxu
ubhengxeshe
ibhinqa
ibhobhile
umbhobho
umbhiyozo
ubhobhoyi
umbhodamo
ubhontsi
icabanga
ucaca
isicaka
icala
icamagu
icandelo
ucanzibe

icawa
icebiso
iceba
icekwa
umcephe
icephe
isicengo
iceya
izicheko
ichele
isichenge
isichenene
ichitywa
amacholi
umcholwa
isichopho
ubuchopo
isichotho
uchuku
ichule
ubuchule
uchulumanco
isichumiso
umchwebelele
ubuchwepheshe
ucikicane
umcinga
ucingo
amacirha
isicithi
icongwane
umcondo
umda
udaba
isidabane
udakada
isidala
umdala
isidalwa
isidamba
isidanga
idangatye
idelakufa
isidenge
isidikimfa
isidima

udlalani
umdlanga
isidlangalala
idlavu
idlelo
umdlezane
udlezinye
idliso
ukudliwa
isidlo
isidlokolo
idlolo
udlomdlayo
umdlungu
udlwabevu
isidlwengu
idobo
ubudoda
isidodo
umdolomba
isidudla
idyudyu
ifanankosi
umfanekiso
umfaneleko
umfazana
isifede
ufefe
isifingo
umfinya
isifinyezo
umfuxane
umfuyi
igaba
umgada
igalelo
umgama
umgaqo
igazi
ugcado
igcisa
ukugxwala
ihempe
ubuhilihili
umhla
ihlaba

umhlabangubo
umhlabeli
umhlakulo
isihlanganisi
isihlangu
uhlaselo
isihlava
umhlekazi
umhlekisi
umhleli
ubuhlobo
isihlomelo
ukwindla
ijaja
ijaji
ijengxeba
ijikazi
ujingi
ujobela
ikamva
ikhaba
isikhaka
isikhakhamela
ikhakhasholo
isikhala
isikhalazo
isikhali
ikhangala
umkhandi
ikhatshu
umkhonzi
ilahleko
ilalela
ilangatya
ilaphu
umlawuli
ubulembu
isimahla
umahlalela
isimamva
isimelabizo
imfeketho
isinciphiso
incindi
umncethezi
inceku

indawo
indebe
indlezana
indlamanzi
indlazi
indwadunge
indwalutho
ingcathu
ingxangxa
ingxangxasi
ingxangxosi
isomelezo
umondli
umonakalo
umongi
umongo
isonka
upapasho
umpapasho
ipasika
ipasile
isipeliti
ipatyutyu
uphahla
isiphako
umphako
umphanda
iphanyazo
isiphango
iqabaza
uqabisisu
iqadi
umqala
iqaqoba
umqaqoba
uqavashe
iqela
iqegu
umqhagi
isiqhazolo
umqhele

irhabasa
irhafu
irhamncwa
irharha
urhexe
irhengqe
irhewu
umrhumo
umrhubhe
isabile
isandle
umsantsa
isamani
iselwa
isishiqi
emthonjeni
entabeni
asingomakhwenkwe
emlanjeni
ngamakhwenke
bantu
asingobantu
ngabantu
asibantu
kubantu
ebantwini
umshologu
isishuba
ishwangusha
ubusika
isisila
usinga
umtha
isitha
umthabalala
isithako
ithambeka
ithambo
imithombo
isithombo
isithonga

ithongo
imvaba
umvambo
uvalo
umvandedwa
ivamna
imvana
imvelo
uvelwano
iliwa
iwaka
umwethu
isiwiliwili
isixando
uxam
ixandeka
ixethuka
umyalelo
umyalezo
iyelenqe
uyihlo
uyihlokazi
uyisemkhulu
uyisemncinci
uyisekazi
isiza
umzabalazo
umzali
umzalikazi
umzalwana
isizamva
isizananina
isizekabani
isiziba
izibazana
izibuko
isizungu