ADDIS ABABA UNIVERSITY
COLLAGE OF NATURAL SCIENCE
SCHOOL OF INFORMATION SCIENCE

Developing a Stemming Algorithm for Awngi Text: A Longest match approach

By

Tsegaye Misikir

A THESIS SUBMITTED TO THE SCHOOL OF GRADUATE STUDIES OF ADDIS ABABA UNIVERSITY IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF MASTERS OF SCIENCE IN INFORMATION SCIENCE

ADDIS ABABA, ETHIOPIA

June, 2013

ADDIS ABABA UNIVERSITY
COLLAGE OF NATURAL SCIENCE
SCHOOL OF INFORMATION SCIENCE

Developing a Stemming Algorithm for Awngi Text: A Longest match approach

By

Tsegaye Misikir

Name and signature of Members of the Examining Board

Name and Title	**Signature**	**Date**
_____Chairperson	_____	_____
_____ Advisor(s),	_____	_____
_____ Examiner,	_____	_____

Declaration

I declare that the thesis is my original work and has not been presented for a degree in any other university.

_____

Tsegaye Misikir

June, 2013

This thesis has been submitted for examination with my approval as university advisor.

_____

Solomon Tereffe (PhD)

June, 2013

ACKNOWLEDGMENTS

First and foremost, I would like to thank the Almighty GOD for giving me the strength and capability to complete this work.

Next, I should like to express the deepest gratitude to my Advisor Solomon Tereffe (PhD) for his invaluable and outstanding guidance, support and help throughout my research.

I would like to express my very special thanks to all staff members of the school of information science for their help and guidance on many occasions. I would also like to express my sincere gratitude to Wollo University for giving me this golden chance to further my studies in this Masters program and to finance me fully for my studies.

Last but never be the least, I would like to thank my beloved families, for their love, believe and support towards my study.

Table of Contents

Appendix

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Stemming is the process of removing the affixes from surface words, without doing complete morphological analysis. Stemming is a procedure to reduce all words with the same stem to a common form. It is useful in many areas of computational linguistics and information-retrieval work. In this paper we present the development of a stemmer for Awngi language that reduces words to their stem. The main objective of this research is to investigate the possibility of applying automatic term conflation and finding the optimal way of developing a stemming algorithm for the language.

We apply longest match approach supplemented by context-sensitive and recoding matching principle. The stemmer is evaluated on Awngi text from three domains; news articles, text books and a dictionary. According to the evaluation of the stemmer, it is concluded that an overall accuracy of 91.41% is achieved which is a very good result as it is the first attempt to develop the algorithm.

As the stemmer is the first kind for Awngi language, 8.59 % error is a number that can be minimized by introducing more rules and exceptional rules. Further research is not only required in the algorithm but also in the morphological structure of Awngi language.

CHAPTER ONE

1 INTRODUCTION

1.1 Background

With the availability of enormous amount of data online, it is very essential to retrieve accurate data for user query. There are lots of approaches used to increase the effectiveness of online data retrieval. The traditional approach used to retrieve data for some user query is to search the documents present in the corpus word by word for the given query. This approach is very time consuming and it may miss some of the related documents of equal importance [**1**]. Thus to avoid these situations, stemming has been extensively used in various Information Retrieval Systems to increase the retrieval accuracy.

Nowadays, the term Information Retrieval (IR), most of the time reflects the automated IR system. It has a wide area which deals with storage and retrieval of any kind of media. Frakes [**2**] said that IR system generally take user query, a formal statement of information needs, as an input and returns related documents as output. Due to the nature of words of natural language that can have varying morphological forms, IR system faces some problems in the process of matching user query to the output document.

Thus, focus of research in IR is on improving accuracy and speed of the system, especially in handling natural language query. According to Frakes [**2**], stemming is a core natural language processing technique for efficient and effective information retrieval. Krovetz [**3**] reported that there is an increase of 15-35% in retrieval performance when stemming is used on short document collections and a more modest improvement for longer documents.

Stemming is a computational process for reducing words to their root (or stem), and it can be viewed as a recall-enhancing device or a precision-enhancing device [**4**].

In Information Retrieval Systems, stemming is used to conflate a word to its various forms to avoid mismatches between the query being asked by the user and the words present in the documents [**1**]. For example if a user wants to search for a document on "How to cook" and submits a query on "cooking" he may not get all the relevant results. However, if the query is stemmed, so that "cooking" becomes "cook", then retrieval will be successful. Stemmers are basic elements in query systems, indexing, web search engines and information retrieval systems.

Natural language texts typically contain many different variants of a basic word [**5**]. Morphological variants are generally the most common, with other sources including valid alternative spellings, mis-spellings, and variants arising from transliteration and abbreviation [**5**]. Stemming solves the problem of words having varying morphological forms by successfully reducing words with the same stem to a common form which is proven Porter and Lovins [**5**] [**6**].

Stemming is also used to reduce the size of index files. Since a single stem typically corresponds to several full terms, by storing stems instead of terms, compression factor of 50 percent can be achieved [**1**].

1.2 Statement of the Problem

There is a need for people all over the World to be able to use their own language when using computers or accessing information on the Internet. This requires the existence of a variety of applications including local language spell-checkers, word processors, machine translation systems, search engines, etc.

Natural language has its own characteristics and features. So, it is quite difficult to follow the same stemming pattern and apply the same stemming rules for all languages. Different prefixes and suffixes, as well as individual exceptions, need special handling and a careful formation of a frame with specific norms, applied on the particular language. There exists a vast literature on the principles, methodologies, and problems involved in the application of stemming algorithms to textual documents written in English, European and Asian languages. However, little attention has been given to stemming of textual data in local languages; Awngi language is one of these.

Most of the languages in Ethiopia are included in the Afro-Asia language family. Of these Ethiopian languages majority are Cushitic and Omotic languages and some are Semitic languages [**7**]. The term "Agew" is used by the people who live in both Ethiopia (Amhara and Tigiray region) and Eritrea. According to the 2007 office of Population and Housing Census Commission of Ethiopia [**8**], there are over 1.5 million speakers of this language.

Awngi is categorized under Cushitic language family. The language plays a crucial role for the people of the region in social, political and economic activities. The Awngi language serves as a medium of instruction in the primary schools, and is also offered as a subject in the junior and secondary schools of Awi zone. The language is widely used in public services (e.g., marketing/shopping) and other events such as religious teachings. As a result,

a significant number of people are able to read and write Awngi. Also, a good sum of computers is being used in offices and educational institutions. Such opportunities open bright future to produce more documents in Awngi language.

Awngi is spoken in Amhara regional state and uses geez writing system. The language uses both kinds of morphologies, i.e. inflectional and derivational. As pointed out by Alemayehu and Willet [**9**], depending on the morphological complexity of a language, both inflectional and derivational morphologies can result in very large numbers of variants for a single word. As a result, word form variations can have a strong impact on the effectiveness of information retrieval (IR) systems and on morphological analysis tools. As Awngi is morphologically complex language, there is thus a need for automated procedures that can reduce the size of a lexicon to a manageable level and improve retrieval performance, and also capture the strong relationships existing between different word forms in the language.

So far, stemming algorithms have been developed for different foreign and local languages such as English [**6**] [**5**], Arabic [**10**], Slovene [**11**], French [**12**], Latin [**13**]), Amharic [**14**], Affaan Oromoo [**15**] ,Tigrigna [**16**] and Wolaytta [**17**]but as the researcher knowledge there has never been any effort made to develop a stemmer for use with Awngi text.

At some point in time, we need a retrieval system for Awngi language text. This requires developing a model on how the derivation of Awngi words takes place; and how semantically related words can be conflated together algorithmically.

Since stemmer is language specific, it is not possible to make use of a stemmer developed for other language to conflate word variants in Awngi text. As far as I know there is no stemmer for use with Awngi text, it makes worth doing a research on developing a stemmer for Awngi text.

1.3 Objective of the study

1.3.1 General objective

- The main objective of this research is to develop a longest match stemmer for Awngi language.

1.3.2 Specific objectives

The specific objectives of the research are:

- To review properties of the Awngi text in order to get familiar with the different aspects of the language
- To Analyze and evaluate existing automatic term conflation methods of local languages for their relevance to Awngi text.
- To construct a list of affixes and stop word list used in Awngi from the corpus.
- To Design stemming algorithm(s) to experiment with Awngi text
- To develop a stemmer that conflate inflectional and derivational affixes of Awngi;
- To evaluate the performance of Awngi stemmer and measure its Effectiveness;

1.4 Methodology

1.4.1 Literature review

In order to gain deep understanding about stemming and stemming algorithm, different stemming algorithms developed for different languages such as English, French, Amharic, Afan-Oromo, Tigrigna and others were reviewed.

To understand the morphology of Awngi, review of works on the language was done by consulting different sources such as textbooks, journals, articles and newspapers. Additional information was also collected through personal discussion with professional experts of the language.

1.4.2 Dataset preparation

A text corpus is one of the resources required in natural language processing researches. Selection of text is, therefore, an important component in developing a stemmer. For the purpose of this research, the researcher used a corpus which can be representative of the language. Sample texts of different disciplines such as textbooks, newspaper and dictionary were collected from different sources such as internet, schools and Awngi educational office.

1.4.3 Developing a stemmer

To build the stemming system, the researcher used python programming, because the researcher is more familiar with this programming language and the feature that the language has for natural language processing. The longest match approach is used as a base for developing the stemming algorithm.

1.4.4 Stemmer Testing

Error counting technique was employed to evaluate the performance of the stemmer. Qualitative analysis is used to see the result of the stemming algorithm. The result is represented in quantitative measures by counting the percentage of correctly stemmed and wrongly stemmed word variants. This figure is used to show the accuracy of the stemmer.

1.5 Scope and Limitation of the study

This study defines the role of the stemming procedure in advanced information retrieval system. Because of the relevance to Awngi language, the retrospective overview of stemming algorithms includes only affix removal stemmers. The structure and complexity of the language determined that the Awngi stemming algorithm comprise suffix removal and prefix removal. The Awngi stemmer was tested and evaluated using different test collections.

The evaluation of the study was cover any comparison of the performance between the Awngi stemming algorithm and other local stemmers because the designed stemming algorithm is the first for Awngi and the grammatical

structure of Awngi is distinct from other local languages for which stemming algorithms have been already applied.

1.6 Significance of the research

Awngi has become a medium of instruction in primary school. As a result text books, reference materials and news papers are compiled using the language. Therefore, during writing the documents in Awngi text editors can use the stemmer in correcting spelling errors.

The development of this stemming algorithm will also be used as an input for further research in the development of other natural language processing areas such as text classification, text summarization, machine translation and others. It will also have great contribution in the development of Awngi information retrieval system.

As Awngi is spoken in Amhara region and uses Geez characters, the development of Awngi stemming will have its own contribution in the development of Amharic-Awngi cross language information retrieval.

1.7 Outline of the thesis

This chapter begins with the discussion regarding the background of the research. This is followed by the problem statement and then the objective of this research and the methodology. Finally, the scope and the significance of the study are described. The outline of the remaining chapters is given below:

CHAPTER 2: Presents the overview of the stemming algorithm where it starts with the overview of word conflation and Stemming Algorithm. This is followed by the discussion on the stemming algorithm approach which includes the Affix Removal, successor variety, table lookup and n-gram. This is followed by review of related work. Then the discussion on the evaluation of stemming algorithm and finally followed by the chapter conclusion.

CHAPTER 3: Discusses the general structure of the Awngi language and starts with an overview of Awngi language and people, which is followed by brief introduction to Awngi character set and basic syllabus structure. Then, detailed description of Awngi morphology as the concept of word formation which is the basis for automatic word stemming is presented.

CHAPTER 4: Presents the stemming algorithm of Awngi with brief introduction followed by data set preparation for the algorithm. The discussion continues with the compilation of Awngi stopword, prefix and suffix lists. What follows is the design of the stemming algorithm with the selected approach and sample rule sets generated from the algorithm. Finally the evaluation of the stemmer was followed.

CHAPTER 5: Presents the conclusion and recommendations. This chapter also includes the summary of contributions and future research works.

CHAPTER TWO

2 LITERATURE REVIEW

According to Frakes [**2**], word conflation is a process of matching morphological term variants. Conflation or reduction of word variants to a single canonical form is used both in document indexing procedures to define most appropriate document content descriptors as well as in information retrieval to determine relevant query terms which match document indexing terms. Therefore, word conflation can be done at indexing time and/or at search time.

Term Conflation can be performed either manually or automatically. Right hand truncation is one of the most often used techniques for manual conflation. Many conventional online systems, i.e., ERIC, STN allow the searcher to truncate query terms by using wildcard characters i. e. asterisk (*). For example, more records on the subject INFORMATION will be retrieved if the initial search term is truncated to INFORM*. However, users often are unfamiliar with the truncation approach. Willett [**18**]stressed that two major problems are associated with the manual right hand truncation:

- Over truncation which means that the remaining stem of a word is too short after truncation;
- Under truncation which causes retrieval of too few related relevant words.

For example, in the case of a user over truncating the word INFORMATICS to INFO, then all words related to INFORMATION and INFORMATICS as well as completely unrelated words will be retrieved. In the case of a word being under truncated, a user will retrieve very few if any relevant word e. g. if the word COMPUTERS is truncated to COMPUTER, than all relevant documents related to COMPUTING and COMPUTATIONAL will not be retrieved. Walker and Jones [**19**] also observed that manual truncation is not often used by users as it

demands certain experience and skills. Therefore, the use of manual word conflation in information retrieval systems and OPACs requires trained intermediaries who can help users to overcome the above mentioned problems.

Automatic term conflation includes special programs called stemming algorithms or stemmers which reduce morphological variants of a word to a one, single form. Lovins [**6**], who designed one of the first automatic word conflation programs, defines a stemming algorithm as a "computational procedure which reduce all words with the same stem to a common form, usually by stripping each word of its derivational and inflectional suffixes" [**6**].

2.1 Types of stemming algorithms

As depicted in figure 2.1, automatic term conflation comprises four different approaches [**1**] which includes

- table lookup;
- successor variety;
- n-gram method;
- Affix removal.

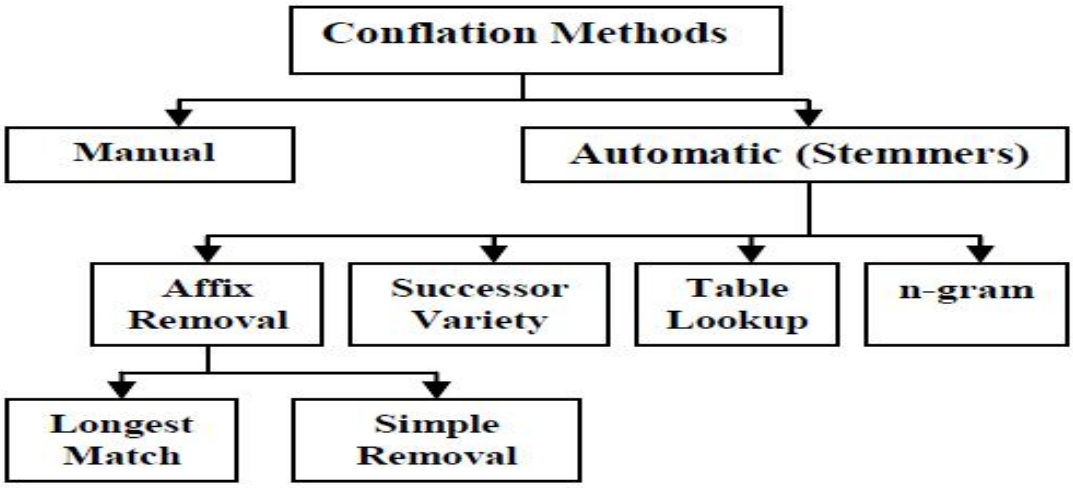

Figure 2.1 word conflation methods

2.1.1 Affix Removal Method

Affix removal is the most often used method, which removes suffixes and/or prefixes from the word so as to convert them into a common stem form. Affix removal method is based on two principles one is iterations and the other is longest match [**1**].

An iterative stemming algorithm is simply a recursive procedure, as its name implies, which removes strings in each order-class one at a time, starting at the end of a word and working toward its beginning. No more than one match is allowed within a single order-class, by definition. Iteration is usually based on the fact that suffixes are attached to stems in a "certain order, that is, there exist order-classes of suffixes. For example, using Porter's stemmer [**5**], which is based on the iterative method, the word ELECTRICITY will be processed in two steps. First of all, the letter Y will be substituted by letter I and after that the ending -ICITI will be transformed to –IC (ELECTRIC), by removing -ITI from the end of the stem.

The longest-match principle states that within any given class of endings, if more than one ending provides a match, the one which is longest should be removed. The first stemmer based on this approach is the one developed by Lovins [**6**]. For example, the Lovins longest match stemmer will remove ending -ICAL from the word ELECTRICAL in one iteration. Comparing with iterative stemmers, the longest match algorithms are often easier to program. However, as longest match stemmers include all compound suffixes, the size of a suffix dictionary is much bigger than it is for iterative stemmers.

2.1.2 Successor Variety Method

Successor variety stemmers [**20**]use the frequencies of letter sequences in a body of text as the basis of stemming. In less formal terms, the successor variety of a string is the number of different characters that follow it in words in some body of text. Consider a body of text consisting of the following words,

back, beach, body, backward, boy. To determine the successor varieties for "battle," for example, the following process would be used. The first letter of battle is "b." "b" is followed in the text body by three characters: "a," "e," and "o." Thus, the successor variety of "b" is three. The next successor variety for battle would be one, since only "c" follows "ba" in the text. When this process is carried out using a large body of text, the successor variety of substrings of a term will decrease as more characters are added until a segment boundary is reached. At this point, the successor variety will sharply increase. This information is used to identify stems.

After the successor varieties for a word has been found, the obtained results are used to segment the word. Hafer and Weiss [**20**]defined four basic strategies for word segmentation:

- cutoff method;
- peak and plateau technique;
- complete word method;
- Entropy method.

The cutoff method segments a word by selecting some cutoff value K. The word boundary (stem and affix) is identified if its successor or predecessor variety reaches or exceeds the cutoff value. The method is easy to implement but it requires selecting of the suitable cutoff value, because if the value is too small, many incorrect cutoffs will be done. If the value is too large, then many correct cuts will be left out.

Using the peak and plateau strategy, the cut in a word is done after the prefix a_i, if only the successor variety Sai >= S_{a-1} and Sai >= $S_{a+1.}$ It means that a segment break is made after a character whose successor variety exceeds that of the character immediately preceding it and the character immediately following it. This method removes the need for the cutoff value to be selected.

The complete word method produces the segmentation after the prefix of a word or before the suffix of the word, if the prefix and/or suffix is a complete word in the corpus. For example, if the test word is ANTIELECTRIC and ANTI appears as a word in the corpus, the break will be made after ANTI.

However, the above mentioned three approaches are based on the variety of successor and predecessor letters, whereas the entropy method uses distribution of those letters in a word. For words with unusually high successor or predecessor counts, i.e., foreign words or abbreviations, the calculation of letter distribution in those words help to avoid segmentation errors. The entropy approach allows the weighting of the importance of each successor and/or predecessor letter in a word by its probability of occurrence.

2.1.3 Table Lookup method

The table lookup method stores all terms and their corresponding stems in a table. Stemming is done via lookups in the table. One way to do stemming is to store a table of all index terms and their stems. Terms from queries and indexes could then be stemmed via table lookup [**20**] by Using B-tree or Hash table; such lookups would be very fast. B-tree is a multi-level tree-structured index, where all associated terms are stored in leaves or buckets. The search for a definite stem can be done by moving down the tree structure and choosing the appropriate branch.

For example, presented, presentable, presenting all can be stemmed to a common stem present. The use of table lookup method has several limitations. One of the limitations is that a table is required for all words and their stems in a particular language. It is unlikely that such a table would contain every word in the language. The second problem with this method is the Storage of this table requires a considerable amount of disk space.

2.1.4 N- Gram Method

Another method of automatic term conflating is called the shared diagram method given in 1974 by Adamson and Boreham [**21**]. A diagram is a pair of consecutive letters. Besides diagrams in this approach, pairs of words are associated on the basis of unique diagrams they both possess. For calculating this association measures we use Dice's coefficient [**2**]. For example, the terms information and informative can be broken into diagrams 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

Thus, "information" has ten diagrams, of which all are unique, and "informative" also has ten diagrams, of which all are unique. The two words share eight unique diagrams: in, nf, fo, or, rm, ma, at, and ti. Once the unique diagrams for the word pair have been identified and counted, a similarity measure based on them is computed. The similarity measure used is Dice's coefficient, which is defined as:

$$S = 2C / (A + B)$$

where *A* is the number of unique diagrams in the first word, *B* the number of unique diagrams in the second word, and *C* the number of unique diagrams shared by *A* and *B*. For the example above, Dice's coefficient would equal (2 x 8) / (10 + 10) = .80. Such similarity measures are determined for all pairs of terms in the database. Once such similarity is computed for all the word pairs they are clustered as groups. The value of Dice coefficient gives us the hint that the stem for these pair of words lies in the first unique 8 diagrams.

Overall, the n-gram method is based on the clustering principle which means that similar words are grouped together. After determination of unique and shared diagrams and computing the similarity coefficient for each pair of words

in the set, these coefficients are used for a clustering algorithm. Automatic word conflation is achieved by considering that all words in a given cluster are equivalent. Stemmed words may comprise different spellings of the same word as well as morphological variants.

The n-gram stemming approach performed good results in particular subject areas i. e. chemistry by testing definite type of words i. e. document titles. For example, Adamson and Boreham [**21**]observed that the n-gram method correctly calculated similarity measures and successfully clustered document titles from Chemical Titles. However, implementation of the n-gram method requires an extremely large amount of computation to cluster any data dictionary [**22**].

2.2 Review of Related work

2.2.1 Review of automatic stemming algorithms for English

In early and mid-eighties construction of stemming algorithms for English language texts and databases reached its highest level of development. As noted by several authors, after this stage any further changes in stemming rules and/or codes will either decrease performance and efficiency of a stemmer or leave it at the same level [**23**].

Martin Porter at the University of Cambridge in 1980 constructed a stemmer which was based on an iterative suffix removal method [**5**]. The algorithm covered a list of term endings and a set of rules including minimum stem length which determined whether the particular suffix could be removed or not.

The principle that vowels and consonants in English terms were arranged in a certain order was built in Porter's stemmer. It meant that all characters could be divided into two groups containing a set of vowels and a set of consonants.

Porter denoted vowels by v and consonants by c. A list of consonants ccc >0 was denoted by C and a list of vowels vvv >0 was marked as V. In that case every term or a part of term can be described in one of the following four forms:

CVCV…. C
CVCV…..V
VCVC…..C
VCVC….V

These forms can be summarized by the expression:

[C] VCVC …..[V] Where the square brackets mean that the presence of C and/or V is optional.

Finally, using m as a measure for the word or a part of the word, the above mentioned formula can be expressed as follows [C] (VC)m [V], where the combination VC repeats m times. The measure m facilitates to determine either the suffix has to be removed or not. According to Porter's stemmer, the case m=0 included the null word, m=1 covers the first word i. e.

m=0 TR, EE, TREE

m=l TROUBLE, TREES

m=2 TROUBLES, PRIVATE.

For example, no suffix would be removed from the term TREE, but ending S would be stripped from the word TROUBLES as in> 1. Porter's algorithm also included several conditional rules which were described in the form SI --S2. It meant that if the term ended with the suffix SI and the stem before SI satisfied the given condition, SI was replaced by S2. For example, if the test word was INFORMATION with Sl=ATION and S2=0, than the algorithm would remove suffix ATION leaving the stem INFORM.

The stemming algorithm operated in five steps. The suffix dictionary included about 60 suffixes which were grouped in five different categories. Step la and lb dealt with plurals and past participles. Step 1 also checks for double consonants in the term endings, except consonants L, S, Z (condition *d not *L

or *S or *Z). Step lc contained condition (*v*) Y-I which changes ending Y to I if m>O, e. g. HAPPY - HAPPI.

Step two, three and four strip suffixes and modified word stems according to the suffix tables, if m>O (in step 4 m>l). For example, for the term ORGANIZATION, suffix -ATION would be removed and the remaining stem ORGANIZ would be modified to ORGANIZE. Step 5a included conditions which dealt with terms ending with vowel E. For example, PROBATE would be transferred to PROBAT but RATE would remain without any changes, because in is less than 1. Step 5b removes double consonants in the remaining stem. Complex suffixes were removed in several steps. For example, for the term OSCILLATORS first of all the ending -S would be removed, leaving OSCILLATOR (step la). After that ending -OR would be replaced by -E i. e. OSCILLATE (step 2). Step 4 would remove suffix -ATE leaving OSCILL. Finally, step 5b stripped double consonant -L leaving the resultant stem OSCIL.

Porter's algorithm was implemented in the experimental information retrieval systems CATALOG and INSTRUCT as well as in an online catalogue OKAPI. As noted by Frakes [**2**], the CATALOG system, which was introduced in 1984, produced good results in information retrieval and provided user friendly front-ends for inexperienced users. The Interactive System for Teaching Retrieval Using Computational Techniques (INSTRUCT) software package was designed in early eighties for students in Library and Information Studies [**24**]. The system covered documents from LISA database for year 1992. The main purpose of the INSTRUCT system was to help the searcher to select the most relevant items which has been identified by the system. Despite Porter's stemmer, which forms the basis of INSTRUCT, having limitations, it produced good results in information retrieval.

Another English stemming algorithm MARS was also designed in the early eighties to provide access to all searchable terms in the database which were

morphologically related to a given search term. The system used linguistic analysis and word decomposition techniques based on morphological lexicon. The MARS stemming algorithm checked each term against the list of stop words and then split terms into prefix, stem, derivational and inflectional suffixes using a morpheme dictionary and morpheme grammar [**25**].

All word stems were grouped together in a stem file where special pointers provided links between text terms and stems to enable successful retrieval of those terms. The morpheme dictionary covered affixes, inflectional endings and fillers, where the morpheme is the longest possible string which was obtained from all possible derivations. The list of morphemes was presented as a tree. For example, the term 'TRADITIONALLY' would be a derivation of 'TRADITION' not 'TRAD (E)'. The morpheme dictionary also included two smaller lists: 'irregular' stems such as Latin and Greek plurals and irregular verb forms; strings which regularly underwent grammatical change i. e. -Y to –IE (ENTRY - ENTRIES).

A pre-processor evaluated whether the string transformations were necessary or not. It was followed by three lists which processed each word using decomposition grammar. A certain stage in a word had to be reached and certain conditions had to be fulfilled to allow the term to be passed to the next stage. All the conditions were listed in the morpheme grammar for the language.

The Paice/Husk stemming algorithm was designed and implemented at Lancaster University in the middle eighties [**26**]. The stemmer was iterative and incorporated one table of rules, where each rule specified either deletion or replacement of an ending. Each line in the rule table included a separate stemming rule. For example, the rule " sei3y> { -ies > -y }" meant that if the word ended in " -ies ", then the last three letters would be replaced by -y i. e.

LORRIES-LORRY (braces cover comments about the action of each rule), and after that the stemmer would be applied again to the stemmed form of a word.

Overall, the stemming algorithm included the following steps:

1) Selection of relevant action, which meant that the final letter of a word or part of a word was checked. If no section of the rules corresponded to that letter, the process was terminated.
2) Testing applicability of the rule. Before applying any of matching rules, a simple acceptability test for each word was carried out. If the final letters of the word did not match the reversed ending in the rule or if the ending matched and the intact flag is set but a word is not intact than go to step 4.
3) Application of the rule;
4) Look for another rule.

The Paice/Husk stemmer has not been formally evaluated, however it works efficiently and is easy to implement.

2.2.2 Construction of Non-English language stemming algorithms

All stemmers described in the previous section have been designed for an English language environment. However, those rules can be applied also to other languages, if the semantic importance of the particular language is based on stems rather than on suffixes. Moreover, as stated above, to date any further developments of English stemming algorithms will not significantly increase effectiveness of information retrieval, whereas improvements can be successfully carried out for a number of more complex non-English languages. Grammatical characteristics and especially morphological complexity determine the adoption of conflation techniques in other languages. For example, it is difficult to apply any English stemming algorithms for other language, as the later may consists of many compound terms. Descriptions of

stemming algorithms which have been designed and implemented for non-English languages e. g. French, Turkish, Slovene, and Latin etc from international languages and Amharic, Oromigna, Tigirigna, Wolaytigna are given below.

2.2.2.1 Stemmer for French terms

French is an inflective language and has a number of irregularities in morphology and orthography. Even the application of the weakest English stemmer for French language will require a comprehensive suffix dictionary of about 3,000 inflectional suffixes. French terms also have differences between linguistic and semantic meanings. According to Savoy [**12**], the stemming procedure for French texts consists of two stages:

1) Morphological analysis of terms;
2) Removal of derivational suffixes according to the grammatical categories.

The morphological analysis requires a dictionary file and a declension file. In dictionary file each term is associated with a certain declension number, gender and grammatical category. For example, the term ROBUSTE (robust) is characterized as adjective, which uses declension number five and the term is masculine in singular form. Declension number five can be found in the declension file which states that ending –s will be removed if the term is in masculine or feminine and in plural form. All declension forms are organized in a truncated digital search tree which determines that the morphological analysis starts from the end of a word. Apart from removing inflectional suffixes, the morphological analysis evaluates the past participle and returns the infinitive form of the verb. For example, the term NEUVES (new) will be processed in following way: first of all three last characters in reverse order i. e. –SEV will be removed from the term (one character at time) and after that character F will be added to the remaining stem NEU forming the stem NEUR
The derivational process is similar to Porter's iterative affix removal approach.

Derivational suffixes can be determined by using a suffix list based on four tables which correspond to four grammatical categories- nouns, adjectives, verbs and adverbs. Each grammatical category covers special rules and several restrictions regarding gender and/or the remaining stem length. When the grammatical category and suffix of the term are determined, it is possible to find a term's stem and the grammatical category of the corresponding stem. For example, for the adjective VOLCANIQUE (volcanic) suffix -IQUE will be removed leaving the stem VOLCAN (volcano) which is a noun.

The French stemmer has been evaluated using three basic tests. The first experiment covered weak stemming which removed inflectional suffixes (plurals, past participle) from 50 test terms. According to results all 50 terms were stemmed correctly. The second test was dealing with prefix removal and the success rate was also high. Finally, the third test which evaluated suffix removal procedure from terms containing only derivational suffixes also revealed high ratio of correct results. It was also observed that the use of grammatical categories can decrease number of over stemmed terms.

2.2.2.2 Slovene stemming algorithm

The Slovene language is similar to English in the sense of creating words by adding suffixes to a basic stem. However, Slovene is an inflective language and covers six different cases where nouns, verbal nouns, adjectives, numerals and pronouns can be not only in singular and plural forms, but also in dual forms. A stemming algorithm for Slovene language was developed by Popovic [**11**]. The stemmer is based on Porter's algorithm and includes a comprehensive list of 5276 Slovene suffixes together with a set of context sensitive rules. Each suffix is associated with a minimum stem length and one of eight codes which determine the definite context sensitive rule that can be applied for the particular term. After the suffix has been removed, three sets o recording rules

check the remaining stem to determine whether it should be modified or not. The stemmer is accompanied by an extensive list of stop words.

Tests and experimental evaluation of the Slovene stemming algorithm were based on the system INSTRUCT, the Sign Test which calculates the probabilistic number of relevant documents, was used to determine the difference between the manual word truncation and automatic stemming.

Results showed that the number of relevant documents retrieved by stemmed and truncated searches were almost equal. However, a further statistical analysis and testing based on the two-tailed Sign Test and Kendall's Coefficient of Concordance showed that the performance between conflated and non-conflated text is far greater in favor of stemming. Moreover, the modified version of Porter's algorithm for Slovene language performed considerably better than the original one.

2.2.2.3 Algorithm for stemming in Latin

Another example of automatic word conflation for non-English language is a stemming algorithm for searching databases of Latin words and texts, which was developed at the University of Sheffield [**27**]. Latin is inflective language which includes five declensions for nouns and adjectives as well as four conjugations for verbs. Because of the complexity of Latin language e. g. many words have more than one distinct stem; manual right hand truncation usually produces poor results. Moreover, users have to have sufficient knowledge of Latin morphological structure not to under truncate or over truncate a search word(s). To overcome this problem, all Latin words were grouped into two separate classes: nouns and adjectives verbs two distinct sets of rules based on the above mentioned classes of words were implemented into the stemming algorithm. The first set of rules removed suffixes from nouns and adjectives in all five declension forms, whereas the second set was stripping suffixes

associated with four conjunctions of verbs. The structured form of the stemming algorithm allows for avoid of proceeding all classes of words through the same stemming routine which means that words with suffixes relevant to nouns would not be processed by the stemming rules for verbs. After the stemming procedure was completed, the algorithm generated two stem dictionaries which included all the resultant word stems.

Analysis and evaluation of the stemming algorithm which was based on several test collections revealed that automatic word conflation for Latin is more efficient than manual right hand truncation. For example, evaluations of the sample test collection C consisting of 49 complete selected documents, in average reached the success rate of 99%.

Besides the above mentioned stemming algorithms for French and Slovene languages, attempts to use automatic term conflation approach have been done also for Turkish, Finnish, Russian and Arabic.

2.2.2.4 Turkish stemming algorithm

Turkish can be characterized as an agglutinative language where words are formed by combining together root terms and morphemes. One of the first parsing algorithms for automatic word analysis was designed by Koksal. The algorithm involves the minimum stem length which is presented in a root dictionary. Each term is processed from the left to right and after a root is determined, the remaining part of the term is searched in a suffix morpheme dictionary to identify morphemes. However, this parsing algorithm did not cover any semantic analysis of terms, which is essential for Turkish and other agglutinative languages, as the most suffixes can be linked only to the limited number of roots. Another drawback of the previous parser was explicit use of iterative procedure for suffix derivation, which is not effective for Turkish, as the number of iterations is not high in this language.

Therefore, Solak [**28**]designed a parser which is mostly based on morphological analysis of the structure of Turkish words. The purpose of this algorithm was to use it as a spelling checker with a further possibility to develop it as a stemming algorithm for information retrieval system.

2.2.2.5 Greek stemming algorithm

Suffix removal algorithm for Greek is one of the first attempts to construct stemming algorithm for language which is based on non-Latin character set [**29**]. The grammatical structure of Greek language covers a rich inflectional system which includes 41 forms of suffixes. Similarly to the above mentioned Slovene, Latin languages, nouns in Greek have four different cases and the declension is carried out according to 41 categories of nouns, e.g., 14 for the masculine, 14 for the feminine and 13 for neutrer. The iterative algorithm was based on two stage suffix removal procedure: analysis and removal of inflectional suffixes; removal of derivational suffixes which correspond to their grammatical categories.

Preliminary evaluation based on two small test collections covering documents in medicine and computing as well as analysis of user enquiries revealed that the majority of errors were caused by under stemming. Although the algorithm have not been implemented into any of Greek databases and/or tested using large document collections, the initial evaluation showed that in 90% of all cases the Greek stemmer produced correct stems.

2.2.3 Review of automatic stemming algorithms for local languages

2.2.3.1 Amharic stemmers

Amharic is a language with very rich morphology and the main previous contribution in the area of stemming Amharic is the work by Alemayehu and Willett [**9**]which investigated the effect of stemming for information retrieval. Nega Alemayehu [**14**] has developed a stemmer for information retrieval purposes. The stemmer has been developed in a manner analogous to that used previously in a Slovene stemmer [**11**].
The algorithm first identifies a set of stop-words and then a set of affixes associated with the remaining content-bearing words. The stemmer removes affixes by iterative procedures that employ a minimum stem length, recording and context sensitive rules, with prefixes being removed before suffixes. Once the stem of the word is obtained, the root is obtained by stripping all the remaining vowels.

Those studies also indicated a positive effect from using the stemmed forms in information retrieval: Alemayehu and Willett [**9**] compared performance of word-based, stem-based, and root based retrieval, and showed better recall levels for stem- and root-based retrieval over word based.

The other Amharic stemmer was developed by Atelach Alemu and Lars Asker [**30**]. The stemmer finds all possible segmentations of a given word according to the morphological rules of the language and then selects the most likely prefix and suffix for the word based on corpus statistics. It strips off the prefix and suffix and then tries to look up the remaining stem (or alternatively, some morphologically motivated variants of it) in a dictionary to verify that it is a possible stem of the word.

The frequency and distribution of prefixes and suffixes over Amharic words is based on a statistical analysis of a 3.5 million word Amharic news corpus. The stemmer had an accuracy of 85% when evaluated on 50 sentences containing 805 words. The stemmer first creates a list consisting of all possible segmentations of the word that is to be stemmed. In a second step, each segmentation is then verified by matching each candidate stem against the machine readable dictionary. If no stem matches the dictionary, the stemmer will modify the stem and redo the matching. If more than one stem matches, the most likely stem will be selected after disambiguating between the candidate stems based on statistical and other properties of the stems. In the cases when exactly one stem matches the dictionary then that segmentation will be presented as the output from the stemmer.

2.2.3.2 Oromo stemmers

One of the stemmers for Oromigna language is the one developed by Wakshum [**15**]. The stemmer uses suffix table in combination with rules that strips off suffix from a given word by looking up the longest match suffix in the suffix list. 342 suffixes are compiled automatically by counting and sorting the most frequent endings. The stemmer finds the longest suffix that matches the end of a given word and removes.

The other oromigna stemmer was developed by Debela Tesfaye and Ermias Abebe [**31**]by considering problems from the previous stemmer developed by Wakshum. The stemmer was adopted from Porter stemmers. Basically, Concepts about measure, arranging the rules in clusters, analyzing word formation based on the nature of their endings are taken from porter algorithm. It is based on a series of steps that each step removes affix byway of substitution rules. These rules only apply when certain conditions hold, e.g. the resulting stem must have a certain minimal length. Most rules have a condition based on the measure. The measure is the number of vowel-

consonant sequences which are present in the resulting stem. This condition must prevent that letters which look like a suffix but are just part of the stem will be removed.

2.2.3.3 Tigrigna stemmers

Girma Berhe made the first attempt to develop Tigrigna stemmer [**16**]which is based on the iterative approach but when it finds two affixes that match with the word, it removes the longest one. The context-sensitive stemmer was considered appropriate because Tigrigna is morphologically complex language. Each affix is accompanied by a context-sensitive rule. The techniques for describing the rules are adopted from Porter [**5**].

The rule for removing an affix is given in the form: Condition A-> SP|S. if a word has affix A and the condition that accompanies it is true, the word will be changed to a stem with pattern SP or the affix is replaced by the string S which could be null(removed) or more.

The stemmer uses five step rules for the purpose of removing affixes. The first step takes the word to be stemmed as an input and removes double letter reduplication. The second step removes prefix-suffix pair by taking the output of the first step as input and checks if the words contains match with any prefix-suffix pair. If the word contains a match and the remaining string has a length greater than the minimum length, then the prefix and suffix are removed from a word. The third step removes prefixes and takes the output of prefix-suffix stripping. The fourth step removes suffixes by accepting the output from the previous stem and checks if the word contains any match from the list of suffixes. If the word has a match and the remaining string is greater than minimum length the suffix is removed from the word. In the last step the algorithm stems reduplication of single letter. This algorithm has recording rule

that is applied after each step is applied for checking some spelling exceptions and making readjustment.

2.2.3.4 Wolayta stemmer

Lemma [**17**]has developed a stemmer for Wolaytta language based on the morphological nature of the language. As stated in his paper Wolaytta is a language dependent on suffixation to form different forms of a given word. Concatenation of suffixes is common in Wolaytta. As a result, two or more suffixes may be concatenated together and attached to a word. In the language, possible list of combination can be very large making difficult to have complete list of combination (concatenations). Besides, concatenation in the language makes suffixes long ones attaching one suffix to another.

Hence, iteratively removing each base suffix one by one is considered as the best choice in this algorithm. As a result of the characteristics of the language, the algorithm adopted iterative approach to develop the stemmer for Wolaytta text. The stemmer developed for stemming Wolaytta text is context sensitive.

In his study, Lemma has employed a semi-automatic means to compile the possible suffix list. In the process of compiling the suffix dictionary, the words in the sample text were first written in reverse order. The reversed list of words was then sorted and frequencies of matching sub-strings identified. Finally the sub-strings which occur more than once are selected as suffixes. First, a word is inputted to the stemmer. The algorithm checks whether a suffix from the list is attached to the inputted word. The suffixes are iteratively stripped from the word and after application of necessary condition, the final word is considered as a stem.

2.3 Evaluation of stemming algorithms

To determine the difference and efficiency between the varieties of term conflation algorithms, several evaluation studies and tests have been carried out for stemmers in information retrieval systems. One of the first experimental studies was done by Salton [**32**] who compared retrieval results based on iterative longest match method using fully stemmed words and terms with suffix 's' removed.

Three document collections i. e. IRE-3 covering 780 computer science abstracts and 34 user queries, ADI consisting of 82 documents and 35 queries and Cranfield-I covering 200 aerodynamics abstracts and 42 queries, have been used for the evaluation study.

Calculations based on 14 dependent variables for each query, i.e. rank recall, log precision, normalized recall, normalized precision and precision for ten recall levels were used to compare both the above mentioned stemming methods. Related group tests and sign tests were used to analyze the calculated data.

For the IRE-3 collection, there were 272 cases, which favored full stemming, in 132 cases the preference was given to suffix 's' stemming and in 72 cases neither one nor another method was preferred. The effect of size for the IRE-3 collection was 175. For the ADI collection, in 254 cases the preference was given to full stemming, 107 cases favored suffix 's' stemming and 129 cases did not favored to any of both stemming approaches. The effect size for ADI collection was 20. For the Cranfield- I collection, full stemming method was chosen in majority cases and the effect size for this collection was 235.

Results revealed that stemming may significantly affect retrieval performance depending on the type of vocabulary i. e. the Cranfield collection is more

technical and homogenous than ADI and IRE-3, therefore the results were better for that collection.

Van Rijsbergen [**2**]evaluated Porter's stemmer against the Dawson's longest match algorithm using the Cranfield-I collection. The results based on ten paired recall - precision levels revealed that Porter's stemmer was slightly better than the Dawson's stemmer.

Another evaluation was carried out by Lennon [**22**] who analyzed the retrieval effectiveness and inverted file compression of several stemming algorithms i. e. RADCOL, Hafer & Weiss, Lovins, INSPEC, Porter. Tests were based on Cranfield- 1400 document collection which covered 1,396 documents and 225 user queries. For each stemmer, words from document titles and user queries were stemmed and stems were replaced by stem numbers for easier processing. The effectiveness of document search was defined by measure E, which can be calculated using the following formula:

$$E = 1 - \frac{(I + b^2)\,PR}{b^2P + R}$$

Where P=precision, R=recall and b measures the relative importance added to recall and precision by the user in a case if relevant documents are retrieved.

The evaluation study covered analysis of the relative effectiveness of stemming vs non-stemming. All stemmers except Hafer and Weiss successor variety algorithm performed better information retrieval results than non-stemmed terms. The relative performance of various stemmers was also evaluated and experiments revealed that there is a little difference using one or another stemming algorithm in terms of retrieval effectiveness.

Walker and Jones [**19**]analyzed Porter's stemmer using an online book catalogue RCL. It was observed that stemming can considerably increase recall. Experiments also revealed that weak stemming does not decrease precision, which strong stemming does.

Weak stemming also performed better document retrieval results for OPACs e. g. OKAPI than strong stemming. Therefore, it was recommended to use weak stemming first and to reserve strong stemming for those cases, when no documents have been retrieved by the weak stemming approach.

Three different stemmers Le SMART based on Lovins' longest match algorithm, Porter's iterative stemmer and 'S' algorithm, which conflates singular and plural term forms, have been evaluated by Harman [**23**]. Experiments were based on IRX system which covered Cranfield- 1400, Medlars and CACM document collections. Recall and precision ratio were analyzed for each group of documents. Tests with the Cranfield collection revealed that stemming did not increase significantly performance. The best result in terms of relevance has been retrieved from CACM collection. The experimental study for evaluation of stemmers involved such methods as:

- reweighting of term expansions;
- selective stemming based on query length;
- Selective stemming based on term importance.

Tests revealed that Porter's stemmer produced more term variants for a given word therefore, expanding the initial query. Lovins algorithm retrieved a larger number of term variants after matching the given root of a term. It was observed that after the stemming process, non relevant documents often received higher ranking scores than relevant items. Overall, the study confirmed that all three stemmers did not evidently improved information performance.

Frakes [**2**]carried on experimental study covering the evaluation of right hand truncation vs. automatic stemming. Results showed that there is no evidential difference between right hand truncation and involvement of stemming procedures. However, as it was mentioned by Harman, stemmed query terms are more convenient for end users than use of truncation and wildcard characters.

Summarizing the above mentioned experimental studies, the evaluation of stemming procedure in information retrieval systems can be based on: comparison of different stemming algorithms, i.e. Harman's study; analysis of use of full word Vs stemmed term, i.e. Lennon's test; comparison of truncation vs. stemming, i.e. Frakes study.

2.4 Summary

Several of recent information retrieval systems, i.e. CITE, MARS, INSTRUCT incorporate one or another of stemming methods, e.g. stemming based on morphological analysis of terms, stemming covering suffix dictionaries etc.

According to Willett [**18**], automatic term conflation in information retrieval systems:

- may reduce the number of distinct terms and therefore the size of dictionary;
- may increase information retrieval effectiveness and particularly then recall ratio as the conflation procedure can easily determine semantically similar terms.

Evaluation studies analyzing stemming techniques and stemmers confirmed that automatic word conflation can improve information retrieval performance. There is also no evidence that stemming can degrade retrieval effectiveness. Tests also determined that stemming results often depends on the type of

vocabulary, involved in a information retrieval process. It was also observed that stemming increases recall ratio but at the cost of decreasing precision.

Overall, at present most research and experiments regarding stemming procedure and stemming algorithms has been carried out in English language environment and based on English materials. Analysis of computerized term conflation in non-English languages reflected that very few stemming algorithms have been developed or adopted for information retrieval purposes in other languages than English. The complexity of language structure is often the main reason for such restrictions. To date no stemming algorithms exist also for Awngi. As the structure of the Awngi language is different from other local and international languages, the automatic conflation methods developed for these languages cannot be used with Agew text. In order to determine the appropriateness of implementation of automatic term conflation for information retrieval in Awngi, it is necessary to characterize the morphological structure of Awngi language.

CHAPTER THREE

3 STRUCTURE OF THE AWNGI LANGUAGE

Awngi is a Central Cushitic language spoken by 1.5 million people in an extensive area in northwest Ethiopia, including all of Awi Zone, but also some areas of the Metekel Zone of the Benishangul-Gumuz National Regional State, and various places in the Alefa and K'wara Woredas of the North-Gonder Zone of the Amhara National Regional State [**33**]. The Alefa and K'wara varieties have sometimes been called Kunfäl, but are dialects of the Awngi language.

The Awis have been granted their own Nationality Zone in the Amhara National Regional State and have decided to establish Awngi as the medium of instruction for primary education. Therefore, orthography was created in the late 1990's. Using this orthography, some textbooks were published and are now used in primary schools.

3.1 The Awngi alphabet

Awngi, similar to Amharic and Tigrigna, uses Geez alphabet. Contemporary Awngi language consists of 35 characters: all the Awngi characters are divided into two basic groups: vowels and consonants [**33**]. There are six vowels in Awngi language.

Vowel ɨ: e.g. ɨsté 'it is called'
ɨngɨr 'back'

Vowel i: e.g. kʰukɾí 'dawn'
tinkʰɪf 'push'

Vowel e: e.g. dek 'well'
sentê 'tear

Vowel a: e.g. tablí 'father'
dad 'street'

Vowel u: e.g. lɨbu ‘slow’
angutʃa ‘cat’

Vowel o: ɨnnoʤi ‘we’
tafo ‘hand’

Awngi language contains the following consonants listed in table 3.1 and several consonants are pronounced as they are spelled.

		Labial	Alveolar	Palato-velar		Uvular	
				Plain	Labialized	Plain	Labzd
Plosive	Voiceless	P, ፕ	T, ተ	K, ከ	k^w, ኩ	Q, ቐ	q^wɨ ቕ
	Voiced	B, በ	D, ደ	g, ገ	g^wɨ ጉ	G, ጎ	G^w, ጎ
Affricate	Voiceless		t͡s, ፀ	t͡ʃ, ቸ			
	Voiced		d͡z , ዘ	d͡ʒ ጀ			
Fricative		F, ፈ	S, ሰ	ʃ ሸ			
Post-stopped fricative			ኦ	ህ			
Nasal		M, መ	N, ነ	ŋɨ ኘ	$ŋ^w$ ኜ		
Flap			R, ረ				
Approximant		W, ወ	L, ለ	Y, የ			

Table 3.1 List of Awngi consonants

3.2 Basic Syllable Structure

The Awngi syllable in most cases fits the maximum syllable template CVC (C = consonant, V = vowel) [**33**]. This means that there is at most one consonant each in the syllable onset and the rhyme. Exceptions to this happen at word boundaries, where one [-sonorant] extrametrical consonant may appear:

ʤɨfint CV-CVCC ‘fear’
ɢsánti CCVC-CV ‘big’

This means that at word-medial there should be no clusters of more than two consonants, which is true in most cases. The most common syllable in Awngi is CV. At word-medial there are no clear examples of vowel clusters. Syllables beginning in a vowel are possible, but only word-initial, as in **asip** V-CVC 'think'. The only vowels acceptable in this position are the central vowels /a/ and /ɨ/.

Therefore, non-central vowels cannot occur at word-initial. There is an issue, however, with word-medial consonant clusters. With some speakers, even combinations of two consonants are broken up by inserting the vowel /ɨ/, like combinations of plosives at differing places of articulation, as in **kágɨtúga** (CV-CV-CV-CV), 'you dried'. Elsewhere, one hears Awngi speakers producing three-consonant-cluster words *without* an epenthetic vowel /ɨ/, when these clusters can be easily pronounced, as in **implá** (VC-CCV), 'one'. Therefore it appears that the number of consonants is not as important as the ability to pronounce a given combination as determined by the sonority hierarchy. Note, however, that the examples given here can be pronounced differently by different speakers or even by the same speakers at different times, as **kágtúga** or **impilá** [**33**].

In spite of this individual variation, the Awngi language usually breaks up CCC-clusters in a word-medial position by inserting the vowel /ɨ/. This is a strong indication for the validity of the maximum syllable template CVC. This maximum syllable template is responsible for certain decisions regarding the interpretation of several sound sequences.

3.3 Morphological system in Awngi language

3.3.1 The concept of morphology and morphemes

Morphology is the study of the structure of words, and of the way in which their structure reflects their relation to other words: both within some larger construction such as sentence and across the total vocabulary of the language [**34**]. Each word consists of morphemes which are the smallest unit of a language that cannot be segmented further and contains a constant meaning [**35**]. Root, suffix, prefix and ending or flexion are the basic types of morphemes.

- root is a morpheme, which contains the meaning of a word;
- ending or flection is a morpheme, which is located at the end of a word and which is flective according to the case and declension of a word;
- suffix is a morpheme, which is located between the root and the flexion;
- Prefix is a morpheme, which is located before the root.

Grammatical concepts, i.e. word-final and stem are also used for the description of a word. The word-final is a part of a word which contains the last suffix together with the flection.

3.3.2 The inflectional system of Awngi words

Awngi is an inflective language and has nine general categories or classes of words [**33**]:

I. nouns (e. g. **አቺ** - a man, **ኽን** -a house);

II. pronouns (e. g. **እንት** - you, **አን** - I);

III. verbs (e. g. **ትንክፍ** - to push, **አንቤብ** - to read);

IV. adjectives (e. g. **አዊ** - sunny, **ድማ** - red, **ሊጊሲማ**-tall);

V. numerals (e. g. **አንኳ** - five, **ሻይ** -a thousand);

VI. adverbs (e. g.**አይኝ** - yesterday, **እንማቺ**- nearly, **ቻ**- tomorrow);

VII. prepositions (e. g. **ሊ** - with);

VIII. conjunctions (e. g. እስታ - and);

IX. particles (e. g. ይኺቸ - only, ሚንች- much);

Nouns, pronouns, adjectives, verbs and numerals form the above listed word classes are categorized under the inflectional words, while adverbs, prepositions, conjunctions, and particles are categorized under the non-inflectional category. Non-inflectional categories or words containing only grammatical morphemes are also called non-content bearing words, which can be considered to be included in a stop word list for an information retrieval system.

3.3.3 Noun morphology

3.3.3.1 Gender

As shown in table 3.2, Awngi has two genders; masculine and feminine. Masculine gender is marked by final /-i/ or a zero morpheme/ Ø /. The feminine is indicated by the ending/-a/. Most nouns referring to objects are masculine while use of feminine gender for these objects has diminutive or derogatory connotation.

	Masculine	Feminine
1	ግሴኛ Giseŋ-Ø ‘dog’	ግሴኟ Giseŋ-a ‘ bitch’
2	ፊሪሲ Firisi ‘horse’	ፊሪሳ Firisa ‘ mare’

Table 3.2 inflection of gender

3.3.3.2 Number

Awngi has two numbers; singular and plural. There is no gender distinction in the plural. The most plural marker is /ካ፣-ka/. It comes right after the stem. Table 3.3 shows plural nouns with /ካ፣-ka/.

	Singular	Plural
1 a.	ታይ Tay 'ram/sheep'	ታይካ Tay-ka 'sheep'
b.	ታያ Taya 'ewe'	
2 a.	ፊሪሲ Firisi 'horse'	ፊሪስካ Firis-ka ' horses'
b.	ፊሪሳ Firisa ' mare'	
3	ልኩ lik^{w} 'leg'	ልኩካ lik^{w} -ka 'legs'
4	እንፀ Instu 'thin'	እንፀካ Instu-ka 'thin'

Table 3.3 inflection of number

As we can see from the gender markers /-i/ and /-a/ are deleted before plural marker /-ka/. In the words, /-ka/ is attached to the stem or base form.

Awngi has some nouns which reduplicate for plural, as ስር sir 'child', ስራስሪ sarasiri 'children' , ኪሲ kisi 'priest' , ኪሳኪሲ kisakisi 'priests', ኹና xuna 'woman' , ኹናኹና xunaxun, women.

All adjectives modifying plural nouns take plural marker suffix/-ka/, with rare exceptions where base forms may optionally be used as plural as shown below.

ሻሳነት-ካ አቃ ይነቱና

Xisant-ka aq yintuna

Elderly people come

3.3.4 Personal pronouns

Like nouns, Awngi pronouns inflect for cases except for nominative which is unmarked. The first and second person singular is suppletive for the oblique case. In the first and second persons of the singular, there are distinct forms for subject and oblique pronouns, but only one form for the rest [**36**]. The subjective forms are /አን *an* /and /እንት *int* /while the oblique forms are /ይ *yi-* / and /ኪ *ki-/* respectively. The later appears only bearing case marking morphemes; they do not appear independent of the markers. Plural pronouns suffix /ስ -s /before they show accusative comitative-directional and genitive cases.

The singular of second and third persons have reverent forms when used to express respect. These are: /እንቱ into/ and /ቻ ŋa / respectively. The following table 3.4 shows Awngi personal pronouns in different cases.

Pronouns			nominative	accusative	Genitive	Comitative	Locative	Ablative	diercti onal	Comitative-directional	Dative
1st person	singular		An	Iy-a/iy-wa	Yi-w	Yi-li	Yi-da	Yi-das	Yi-so	Yi-wla	Iya-s
	Plural		innoji	Innoji s-o	Innoj is-u	Innoji-li	Innoj ida	Innoj i-das	Innoji-so	Innoji-wla	Inn oji-s
2nd person	Singular	Non-reverent	Int	K-wa	k-u	Ki-li	Ki-da	Ki-das	Ki-so	K-ual	Ku-s
		reverent	Intu	Int-o	Int-u	Intu-li	Intu-da	Intu-das	Intu-so	Intu-la	Intu-s
	Plural		innoji	Innoji-o	Innoj i-u	Innoji-li	Innoj i-da	Innoj i-das	Innoji-su	Innoji-wla	Inn oji-s
	Singular	Non-reverent	Ɗi	Ɗe-wa	Ɗi-w	Ɗi-li	Ɗi-da	Ɗi-das	Ɗi-so	Ɗi-wla	Ɗi-s
		reverent	Ɗa	Ɗa-wa	Ɗa-w	Ɗa-li	Ɗa-da	Ɗa-das	Ɗa-so	Ɗa-wla	Ɗa-s
	Plural		ŋaji	Ɗajis-o	Ɗaji-u	Ɗaji-li	Ɗaji-da	Ɗaji das	Ɗaji-so	Ɗaji-ula	Ɗa-s

Table 3.4 inflection of personal pronouns

3.3.5 Cases

Case is the syntactic feature that defines relation between noun phrases and their predicates and it applies to all phonetically overt noun phrases [**37**] . Case is the system of marking dependent nouns for the type of relation they bear with their heads. The verb is taken to be the head of the clause since it largely determines what dependants may be present.

3.3.5.1 Accusative case

Accusative case in Awngi is expressed by inflectional elements and supra-segmental feature. The inflectional elements are/-o/,/-wa/,/-e/ and /-sa/.

Let us see the following examples

A) **ግሴኟዋ ኩና**
Giseŋa-wa ku-na
They killed the bitch

In the above example (A), the accusative case marker/-wa/ is suffixed to Giseŋa 'bitch' to mark case. The occurrence of /-o/, /-wa/ and/-e/ is phonologically conditioned. /-o/ occurs after consonants or/u/; /-wa/ occurs following/a/ or/-e/ and /-e/ occurs with nouns ending in /i/ which deletes after affixation of [-e].

B) **ግሴኝ ኩና**
Giseŋ-o ku-na
They killed the dog

C) **ግሴኝ-ካ-ዋ ኩና**
Giseŋ-ka-wa kuna
They killed the dogs

D) **ዛግሬ ኩና**
Zagri +-e zagr-e ku-na
They killed the monkey

As can be seen from the examples (B, C, D), the accusative marker is/-o/ in nouns that end in consonant, /-wa/ that end in a vowel/-a/ and /-e/ in nous that end in /-i/.

The occurrence of /-sa/ is grammatically conditioned in that they occur with possessive forms that serve as qualifiers and with verbs of relative clauses. The following examples (E, F) shows the accusative case marker for/-sa/

E) **ይውሳ ፍያሎ ኩና**
Yi-w-sa fiyal-o ku-na
They killed my goat

F) **ይትሳ ፍያላዋ ኩና**

Yit-sa fiyala-wa ku-na

They killed my female goat

3.3.5.2 Dative case

Dative is typically the case of indirect object [**38**]. In Awngi, dative case is marked by the suffix/-s/ that occurs attached to the indirect object, which comes following the direct object and before the verb.

Consider the following examples (G, H)

G) **ገንዜቦ ኹናስ ይትኾማ**

Genzab-o xuna-s yitix^{w}a-ma

Did u give the money to the woman?

H) **ገንዜቦ ፉቻስ ኹናስ ይትኾማ**

Genzab-o fuca-s xuna-s yitix^{w}a-ma

Did u give the money to the white woman?

In (G), The dative marker (-s) is suffixed to the head noun xuna 'women' but in (H) it is suffixed to both the qualifier and the head which is not common.

3.3.5.3 Comitative Case

Comitative case indicates the accompaniment in action [**38**]. In Awngi, the suffix /-li/ is expressed this case.

Consider the following examples (I, J)

I) **ታብሊ ጀርሊ ይንትኺ**
Tabli jer-li yintix^{w}a
The father came with his son

J) **ፉቻ ኹናሊ ካስኾ**
Fuca xunna-li kasix^{w}a
He went with the white woman

As shown in (I, J), /-li/ is suffixed to the noun jer and xuna.

3.3.5.4 Genitive case

Blake [**38**] describes the genitive case as encoding the abdominal relation that subsumes the role of possessor, and the label possessive case as a common alternative.

The occurrence of Awngi genitive marker is very complex. They vary in accordance with not only the number and gender of the possessed nouns but also for phonological reasons. The possession suffixes are /-u/,/-t/,/-ti/,/-ku/, /-kw/ and /-su/.

- /-u/ when the possessed noun is singular masculine.
- /-t,-ti/ when the possessed pronoun is singular feminine.
- /-ku,kw/ when the possessed noun is plural
- /-su/ after the plural pronoun but only when the possessed is singular.

Consider the following examples (K, L, M, N, O, and P)

K) አቓው ቢሪ

Aqqa +u Aqqa-w biri

The woman's ox

L) አቒት ኣሏ

Aqqi-t illwa

The man's cow.

M) Aq-ti illwa

The men's cow.

N) አቓካው ፊየልካ

Aqqa-kw fiyala –ka

The woman's goats

O) አቕኩ ፍያልካ

Aq-ku fiyala –ka

The men's goats

P) እኖጂሱ ኞን

Innji-su ŋin

Our house

The phonological conditioning concerns /-u/ and/-w/, /-t/and/-ti/ and /-ku/ and/-kw/. /-t/and /-kw/ occur following a vowel whereas /-ti/ and/-ku/ occur following a consonant. The occurrence of suffix /-su/ is limited to plural pronouns.

3.3.5.5 Local cases

Local cases express notations of location ('at'), destination ('to'), source ('from') and path ('through') [**38**]. The distinguishable local cases in Awngi are locative, ablative, directional, directional-comitative, and purposive. Locative case shows location and shown by suffix /-da/ which is suffixed to a word referring to a noun. Ablative case express the origin and is shown by suffix /-das/. Directional is self explanatory and is expressed by suffix/-so/. Directional-comitative shows not only direction but also accompaniment in that something or someone has gone or been taken to some other person to live or stay with them and is shown by suffixes /-wla/,/-ula/ and /-sula/.

Consider the following examples (Q, R , S, T, U)

Q) ኾና ክንዳ ዝኬ

Xuna ŋin-da zike

The woman is in the house

R) ኾና ክንዳስ ቲንትኺ

Xuna ŋin-das tintixwa

The woman came from the house

S) ኾና ክንሶ ካትኾ

Xuna ŋin-so katixwa

The woman went towards the house

T) ኺ ታላሱላ ካታ

Ɔi-tala-sula ka-ta

She has gone to her father

U) ብና ካታ

Bin-a ka-ta

She has gone to river (for fetching water of washing clothes).

3.3.6 Verb morphology

In Awngi, there are two clearly identified aspect of a verb; perfect and imperfect [**36**]. As aspect is a term that covers how we view an event within a time frame. The perfect aspect refers to temporally bounded situation whereas imperfect aspect refers to a situation which is not temporarily bounded but which is rather an explicit reference to its internal structure. Hence, perfect aspect is very often related to past tense while imperfect aspect refers to habitual, continuity, progressivity, and etc. table 3.5 shows the perfect and imperfect forms of the verb **ሱግ** sug 'pound'.

Person			Verb aspect		Aspectual suffixes		
			Perfect	imperfect	perfective	imperfective	Jussive
1st	Singular		Sug-ixʷa	Sug-a	- ixʷa	-á	Sug-is
	Plural		Sug-n-ixʷa	Sug-n-a	- ixʷa	- á	Sug-n-is
2nd	Singular		Sug-t-ixwa	Sug-t-a	ixʷa	- á	Sug-t-is
	Plural		Sug-t-un-a	Sug-t-a	-a	- á	Sug-t-in-is
3rd	singular	masculine	Sug-ixwa	Sug-a	- ixʷa	- á	Sug-is
		Feminine	Sug-t-ixwa	Sug-t-a	- ixʷa	- á	Sug-t-is
	Plural		Sug-un-a	Sug-an-a	- á	- á	Sug-in-is

Table 3.5 inflection of verbs

3.3.7 Morphology of numerals

Numerals are those items that include the cardinals (one, two, and three…) and the ordinals (first, fifth…). The process of forming ordinals from the cardinals is as follows:

ላኸ - one	እምፐላንቲ - first
ላኃ - two	ላኃንቲ - second
ሹኻ - three	ሹካንቲ - third
ሴዛ - four	ሴዛንቲ - fourth
እንኳ - five	አንኳንቲ - fifth
ዋልታ - six	ዋልታንተ- sixth

The morpheme of the ordinal marker is /ኣንቲ/ which is suffixed to cardinal numbers ending in a consonant and /ንቲ/ is suffixed to those that end in a vowel /ኣ/.

3.4 Summary

Morphological structure of Awngi language shows that both the inflectional and derivational morphologies involve suffixing. Analysis of the Awngi language and grammatical structure revealed that words are created by adding suffixes to a basic stem. It is also shown that the language uses an extensive concatenation of suffixes. This characteristic of the language make a very short stem into a long word. As it is evidenced by different authors, such nature of the language indicates the complexity of the language's morphology. The complexity of the language is one of the main reasons for a language to desire a stemmer since stemmer is an automated mechanism that conflates variants of words. It is an important tool to access documents in natural language text

CHAPTER FOUR

4 CONSTRUCTION OF AWNGI STEMMING ALGORITHM AND EXPERIMENTATION

4.1 Introduction

The grammatical structure of the language is one of the basic factors which determine the choice of appropriate stemming algorithm for automatic word conflation. As presented in section two, the majority of English and non-English stemming algorithms are based on either longest match or iterative stemming principle. Analysis of morphological structure of Awngi language shows that suffix removal algorithm is the most relevant stemming needs to be applied.

Taking into account the requirements for automatic word conflation in the Awngi language discussed previously, the longest match principle for stemming algorithm was chosen as the basis for the development of Awngi stemmer.

4.2 Data set

In order to develop and test the stemmer, there is always a need to have sample text data in the specified language. Since there is no document for Awngi language available like the TREC and the CLEF collections for the English language, we used our own collections. Therefore, the sample dataset used for this research is obtained from Awngi dictionary, from different teaching and reference books and Awngi newspaper that are found in Awi zone. In terms of number of words, the sample text consists of a total of 6515 words.

4.2.1 Test set

To test the performance of the stemmer, 20% (1303 words) of the sample text were used as a test set (or data). First the entire words in the sample text are put in alphabetic order and then systematic random sampling technique is employed to select the words that have to be included in the test set. To come up with the test set, the first word was deliberately selected and then every 5th word is selected. The number of words to be passed before getting the next word to be selected was decided by dividing the size of the sample text to the number of words to be included in the test data. The aim in doing this was to get whole test data in one pass through the sample data without coming back for another round in order to protect the chance of a word to be selected again and again.

4.2.2 Training set

All the 5212 words (80% of the sample test) that were not included in the test set were used for training the stemmer.

4.3 Compilation of Awngi affix list

Among local languages for which extensive list of prefixes, suffixes, and prefix-suffix pairs are reported are Amharic [**14**] and Tigrigna [**16**]. But from the literatures which we have reviewed, Awngi is the language which is dependent on suffixation with only four prefixes. The morphemes that are used to represent a prefix and prefix-suffix pairs in other languages are almost represented by a suffix in the case of Awngi. For example, in Amharic **የ-አበበ** the prefix (**የ**) is prefixed to the noun **አበበ**. This is equivalent with **አበበ-ስ** where **-ስ** is suffixed to the noun. The same is true for others.

4.3.1 The compilation of prefix lists

The prefixes that are used to develop the algorithm were compiled from different sources based on the grammatical functions of the affixes and their occurrence frequencies among the Awngi words found in the document collection.

ኺ
ይው'
ኩው
ኺው

Table 4.1: list of prefixes obtained from sample text

4.3.2 The compilation of suffix lists

The same procedure which is used to compile prefix is employed in order to compile the possible suffix list (word endings). Sample suffixes are shown in table 4.2 but the whole list is given in the appendix I.

ሊ	ውሊ	ው'
ዋ	ት	ዳ'
ሱ	ዳስ	ስ'
'ንስ	ትስ	ትንስ' '
ኒደስ	ዋላ	ውላ
ኩሳ	ውኩሳ	ውሳ'
'ካውሳ	ዋስታ	ታና'
'ና	ኻ	ኻስ

Table 4.2: list of suffixes obtained from sample text

4.4 The algorithm

The stemming algorithm focuses on the removal of prefixes and inflectional suffixes for any given inflectional Awngi word. Taking into consideration the Awngi grammar and sample text, the researcher has identified 165 different suffixes for the 6 main inflectional word types: noun, adjective, gender, pronouns, number and verb. As the Awngi stemmer cannot distinguish

between the types of the words, the approach is straightforward for the removal of prefixes and suffixes. The main idea is to filter the word through a prefix list and a suffixes list which contains all the possible endings.

An algorithm is developed based on the longest-match, context-sensitive and recoding matching principle which uses more than one order-class. The obvious disadvantage of this method is that it requires generating all possible combinations of affixes. A second disadvantage is the amount of storage space the endings require. The first disadvantage may also be present to a large degree when one is setting up an iterative algorithm with as many order-classes as possible. To set up the order-classes, one must examine a great many endings. Furthermore, it is not always obvious to which class a given string should belong for maximum efficiency. It is also entirely possible that the occurrence of members of some classes is context dependent. In short, while an iterative algorithm requires a shorter *list* of endings, it introduces a number of complications into the preparation of the list and programming of the routine.

Context sensitive is used in order to get the best results; certain endings should not be removed in the presence of certain letters in the resultant stem, usually those letters that immediately precede the ending. In order to control derivational changes of alphabets after removing suffixes, recoding matching is used. Recoding occurs immediately following the removal of an ending and makes such changes at the end of the resultant stem as are necessary to allow the ultimate matching of varying stems.

The following is the longest match algorithm developed to conflate word variants

1. Get the word to be stemmed
2. Determine where to begin the ending list
3. Search ending list for match to the last part of the word being stemmed

   If ending found

   Check for context sensitive rule satisfied if any

   Remove endings

   Else

   Go to 4
4. Search a list of transformation for a match on remaining

   If a match found

   Recode a stem according to the rule
5. If end of file not reached

   Got to 1

   Else

   Stop processing
6. Output stem

The way the algorithm works is explained using example as follows. First, a word is passed to the stemmer. Let the stemmer get the word **ክንቲኒዴስ**. In this word, we get three suffixes {-**ኒ**}, {-**ዴ**}, *and* {-**ስ**}. The stem is **ክንት** 'education'. After getting the word **ክንቲኒዴስ**, the suffix file is opened to determine where on the ending list to begin. Assume that the **ኒዴስ** suffixes listed above are kept in the suffix dictionary in the longest match suffix list. When reading suffix from the suffix dictionary, the stemmer gets the suffix {-**ኒዴስ**}.while scanning list of suffixes it may get suffixes like -**ዴስ**. As the stemmer is longest match and context sensitive it will check the rule if any and then it will immediately remove only -**ኒዴስ** resulting with the word **ክንቲ**. Then, this leads to the fifth step where the resulting stem **ክንቲ** will be in the list of transformation for a match.

The stem for ክንቲ is ክንት , therefore the transformation rule will be applied to the remaining word.

After application of necessary condition, if any, on the finally identified stem the word is considered as a stem and recorded in the stem dictionary. The procedure continues reading next word, EOF not reached, for the same stemming task.

4.5 The rules

Trying to deal with each suffix individually, we have created a decentralized algorithm. The rules sets are presented in appendix II

4.6 Evaluation of the stemmer

As it has been pointed out by Hull [**39**], the most often used methodology for the evaluation of a new information retrieval strategy, i.e. stemming incorporates the following steps:

- choice of relevant test collection(s);
- Testing of the new method by carrying out an information retrieval experiment and comparing results, using a baseline, which is obtained from a standard approach, e. g. manual truncation;
- Calculation of traditional evaluation measures, i.e. precision and recall.

The superiority of the new information retrieval technique is justified if it achieves better results than the standard baseline.

The above mentioned approach to information retrieval evaluation is common, because it provides an objective means of evaluation and requires a minimum of experimental work, if relevant test collections are available. The comparison of traditional manual word truncation versus automatic word stemming is one of the generally used evaluation procedures for stemming algorithms [**2**]. Even if the described methodology has been successfully applied for the analysis and

evaluation of both English and non-English languages, it is not used as the base for evaluation of Awngi stemmer.

In Awngi stemming algorithm, the quality of the stemmer is assessed by counting the number of identifiable errors during the stemming process. These errors can be identified by two ways; over stemming and under stemming. When a term is over stemmed, too much of it is removed. Over stemming can cause unrelated terms to be conflated. Under stemming is the removal of too little of a term. Under stemming will prevent related terms from being conflated.

Evaluation of the Awngi stemming algorithm is based on the following test collections:

- an electronic dictionary of Awngi nouns, adjectives, verbs and adverbs in their standard forms;
- Text fragments extracted from different Awngi texts such as text books and newspapers.

Therefore, a good stemmer should obviously produce as few over stemming and under stemming errors as possible. As it has been stated above, the evaluation of Awngi stemmer is done by counting these errors from the sample texts. In this evaluation a correctly stemmed word is considered as any word without prefixes and suffixes attached.

Lastly, a series of experiments were conducted to assess the overall performance of the proposed method. The implemented method was run on different data sets to be evaluated. The data sets were randomly extracted from the Awngi texts created for the development of the stemmer.

Unstemmed word	Expected stem	System output	Over/under
ቲሪትቻንኩ	ቲሪት	ቲሪት	
ጌምናኒ	ጌም	ጌምን	under
አግፅናው	አግፅ	አግፅን	under
አስሜምቻፃው	አስሜም	አስሜም	
ታምናና	ታም	ታም	
አሜታዴስ	አሜት	አሜት	
ጉሽታንኻስ	ጉሽ	ጉሽታንኻ	under
እስካውካ	እስካዊ	እስካ	over
ቲሪትቻፂኩሳ	ቲሪት	ቲሪትቻፅ	under
እንየው	እንየው	እንየ	over
ቸፅቻፃኒስ	ቸፅቻ	ቸፅቻፃኒ	under
ታብሊ	ታብሊ	ታብ	over
ይንታው	ይንታ	ይን	over
እርዳትካማ	እርዳት	እርድ	over
ጌሌፅካ	ጌሌፅ	ጌሌ	over
ዴሜካማ	ዴሜክ	ዴሜ	over

Table 4.3 Sample result of the stemmer

The test result shows that 4.68% (49 words) are under stemmed and 3.91% (41 words) are over stemmed which make the accuracy 91.41%. The distinct stems after conflation are 695. Accordingly, the dictionary size of the test set is compressed by 46.6%.The reasons for under stemmed words are the fact that other few additional suffixes (suffixes not listed in the suffix dictionary) are identified.

In general, reasons for the under stemming and over stemming problems are:

- It was difficult to come up with the complete list of suffixes because of the complexity of the language.
- More conditions/rules are required based on a detailed study of the morphology of the language.

CHAPTER FIVE

5 CONCLUSION AND RECOMMENDATION

5.1 Conclusion

Natural language texts typically contain many different variants of a basic word. Stemming solves the problem of words having varying morphological forms by successfully reduce words with the same stem to a common form.

The aim of this study was to examine the working of automatic word conflation and implementing one of the stemming algorithms for Awngi language. Analysis of the Awngi language and grammatical structure revealed that words are created by adding suffixes to a stem. This statement justified the selection of affix removal algorithm as the basis for the Awngi stemmer. Therefore, in this study the longest-match context-sensitive and recoding matching principle was implemented among the different stemming algorithms.

The testing and examination of the Awngi stemming algorithm was using an electronic dictionary of Awngi nouns, adjectives, verbs and adverbs, as well as fragments of Awngi texts , confirmed that the Awngi stemmer stems both Awngi words in standard forms and in declensions correctly and leaves relevant resulting stems of words as well as removes appropriate stopwords.

According to the evaluation of the stemmer, it is concluded that an overall accuracy of 91.41% is achieved which is a very good result as it is the first attempt to develop the algorithm. The proposed system also generates some errors. The errors are analyzed and classified into over stemming and under stemming. The error rate is 8.6%.

Conflation algorithms have inherent limitations and certain linguistic problems that are common to all conflation algorithms, irrespective of their ultimate use. These error types are inherent to the suffix-stripping method.

5.2 Recommendation

The research work is a prototype stemmer for Awngi language that seems work with a very good accuracy. Like any other prototype, the stemmer has its own drawbacks and further works and improvements are required to improve the efficiency and effectiveness of the stemmer.

As the stemmer is the first kind for Awngi language, 8.6 % error is a number that can be minimized by introducing more rules and exceptional rules and it is our believe that it should be improved by further research to attain more performance. Further research is not only required in the algorithm but also in the morphological structure of Awngi language.

All the rules described in this work can be a base for further research and it can support to develop extended stemming rules covering most of the terms in the Awngi language.

Moreover, the stemmer has to be tested with large amount of texts to prove its real performance. To succeed in this regard we need to apply the Awngi stemmer in a web search engine, which retrieves information from Awngi texts. Then we can have a complete view of the stemming system and the returned results after every search request. If we do so, the evaluation test will be extended into precision and recall.

Finally, this thesis concentrates on longest match stemming algorithm. Other algorithms can be implemented and the performance between them can be compared over a larger collection of data.

6 Reference

[1] Deepika Sharma, "Stemming Algorithms: A Comparative Study and their Analysis," *International Journal of Applied Information Systems*, vol. 4, no. 3, pp. 1-6, 2012.

[2] W. B. Frakes, "Stemming algorithms. In Frakes: ," in *Information retrieval: data structures and algorithms.*: Prentice-Hall, 1992, pp. 131-160.

[3] Robert Krovetz, "Viewing morphology as an inference process," *Artificial Intelligence*, vol. 118, no. 1-2, pp. 277–294, 2000.

[4] Al-Shammari and Eiman Tamah, "Towards an error-free stemming," in *IADIS European Conference Data Mining*, 2008.

[5] M.F. Porter, "An algorithm for suffix stripping," *Program: electronic library and information systems*, vol. 14, no. 3, pp. 130-137, 1980.

[6] Julie Beth Lovins, "Development of a stemming algorithm," *Mechanical Translation and Computational Linguistics*, vol. 11, no. 1 and 2, 1968.

[7] Bender M. Lionen, *Languages in Ethiopia*. london: Oxford University Press, 1976.

[8] "Population and Housing Census of Ethiopia,Summary and statistical report," Addis Ababa, Ethiopia., 2007.

[9] Alemayehu Nega and Willet P., "Stemming of Amharic words for Information Retrieval," *In Literary and Linguistic Computing, vol 17 no 1*, pp. 1-17, 2002.

[10] M. W. Al-Kharashi and Evans, "Comparing Words, Stems, and Roots as Index Terms in an Arabic Information Retrieval System," *Journal of American Society for Information*, vol. 45, no. 8, pp. 548 - 560, 1994.

[11] Popovic, Mirko and Willett, Peter., "The Effectiveness of Stemming for Natural-Language Access to Slovene Textual Data," *In Journal of American Society for Information Science*, vol. 43(5), pp. 384 - 390, 1992.

[12] Savoy, Jacques., "Stemming of French Words Based on Grammatical Categories," *Journal of American Society for Information Science* , vol. 44, no. 1, pp. 1-9, 1993.

[13] Mark Greengrass ,Alexander M. Robertson and Peter Willett Robyn Schinke, "A Stemming Algorithm for Latin Text Databases," *Journal of Documentation.*, vol. 52, no. 2, pp. 172 - 187, 1996.

[14] Nega Alemayhu, "Development of a Stemming Algorithm for Amharic Language Text Retrieval. Ph.

D Thesis," *University of Sheffield* , 1999.

[15] W. Mekonen, *Development of stemming algorithm for Affan Oromo language text, MSc thesis faculty of informatics, Addis Ababa University, Addis Ababa.*, 2000.

[16] Girma Berhe, "A Stemming Algorithm Development for Tigrigna Language Text Documents. MSc Thesis. (Unpublished.)," 2001.

[17] Lemma Lessa, "Development of stemming algorithm for wolaytta text," Addis Ababa, 2003.

[18] Willett Peter, "Automatic indexing of documents and queries," *Document retrieval systems*, vol. 21, no. 2, pp. 4-9, 1988.

[19] Walker, Stephen & Richard M. Jones., "Improving subject retrieval in online catalogues.," in *Stemming, automatic spelling correction and cross reference tables*. London: The Polytechnic of Central London, 1987, p. 2.

[20] M. Hafer and S. Weiss, "Word Segmentation by Letter Successor Varieties," *Information Storage and Retrieval*, vol. 10, no. 371-385, 1974.

[21] G. Adamson and J. Boreham , "The Use of an Association Measure Based on Character Structure to Identify Semantically Related Pairs of Words and Document Titles," *Information Storage and Retrieval*, vol. 10, pp. 253-260, 1974.

[22] Lennon, M., Peirce, D.S., Tarry, B.D. and Willett, P., "“An evaluation of some conflation algorithms for information retrieval”," *Journal of Information Science, Vol. 3 No.4*, 1981.

[23] Harman, Donna, "How effective is suffixing? ," *Journal of the American Society for Information Science*, vol. 42, no. 1, pp. 7-15, 1992.

[24] Hendry, Ian G., Peter Willett & Frances E. Wood, "INSTRUCT: a teaching package for experimental methods in information retrieval. Part 1," vol. 20, no. 3, pp. 245-263, 1986.

[25] Niedermair, G. Th. G. Thurmair & 1. Bfittel. MARS, "a retrieval tool on the basis of morphological analysis," *Research and development in information retrieval*, pp. 375-378, 1984.

[26] Chris D. Paice, "Another stemmer," *ACM SIGIR Forum*, vol. 24, no. 3, pp. 56-61, 1990.

[27] Mark Greengrass ,Alexander M. Robertson and Peter Willett Robyn Schinke, "A stemming algorithm for Latin text databases," *Journal of Documentation*, vol. 52, no. 2, pp. 172-187, 1996.

[28] Solak, Aysin & Kemal Oflazer, "Design and implementation of a spelling checker for Turkish," *Literary and Linguistic Computing*, vol. 8, no. 3, pp. 113-124, 1193.

[29] Kalamboukis, T. Z., "Suffix stripping with Modern Greek," *Program*, vol. 29, no. 3, pp. 313-321, 1995.

[30] Argaw, A. and A Asker, L., "An Amharic stemmer: Reducing words to their citation forms.," in *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, Prague, 2007.

[31] Debela Tesfaye, Ermias Abebe, "Designing a Rule Based Stemmer for Afaan Oromo text," *International Journal of Computational Linguistics (IJCL*, vol. 1, no. 2, 2010.

[32] Salton, Gerard, "Automatic information organization and retrieval," pp. 280-349, 1968.

[33] Andreas Joswig, "The phonology of Awngi ," *SIL International*, 2010.

[34] Anderson, Stephen A., "Morphological theory. In: Frederick J. Neumeyer, ed. Linguistic theory. Part of Linguistics," *the Cambridge survey*, vol. 1, p. 146, 1988.

[35] Parker, Frank., "Linguistics for non-linguists," p. 66, 1986.

[36] Hetzron, Robert, "The Nominal System of Awngi (Southern Agaw)," *School of Oriental and African Studies*, 1978.

[37] Noam Chomsky, *Language and the Study of Mind.* tokyo: Language and the Study of Mind., 1982.

[38] barry. J. blake, *case*. cambridge: cambridge university press, 2001.

[39] Hull, David A, "Stemming algorithms: a case study for detailed evaluation.," *Journal of the American Societyfor Information Science*, vol. 47 , no. 1, pp. 70-72, 1996.

[40] Fox, Christopher, "Lexical analysis and stoplists. In: William B. Frakes & Ricardo Baeza-Yates," *Infonnation retrieval: data structures and algorithms*, p. 113, 1992.

Appendix I: compiled suffixes

ታንኻስ
ኒዴስ
ካዴስ
ስኻስ
ኻስ
ካስ
ኑስ
ኺስ
ውካ
ካ
ፅካዋ
ንትካ
ካዋ
ኹዋ
ዳ
ካዋስታ
ኩሳስታ
ዋስታ
ሚስታ
ኻፂታ
ኻታ
ታንኩሳ
ውኩሳ
ሚውሳ
ናውሳ
ኩሳ
ውዴስ
ፅኻስ
ትንስ
ትስ
ዴስ
ኑስ
ስ
ኃንትካ
ዋ
ስቲካማ
ኻፅካማ
ትካማ
ካማ
ማ
ንኹስታ
ካዳስታ
ካውስታ
ስስታ
ዳስታ
ኺስታ
ታ
ኻፅኩሳ
ውሳ
ኹሳ
ሳ
ኹሳ
ፅንስ
ኻኺስ
ዳስ
ንስ
ስቲካ
ኻፅካ
ፅካ
ኃውዳ
ካኻዳ
ዋዳ
ውዳ
ካዳ
ኻዳ
ዴስስታ
ካስስታ
ካስታ
ውስታ
ስታ
ኙንኩሳ
ካውሳ
ኑሳ
ላንቲ
ስታንቲ

ዋንቲ
ፃንቲ
ጃንቲ
ንቲ
ቲ
ስቱን
ችኽ
ኼኽ
ቄኽ
ዳኽ
ኽ
ሊ
ታኪ
ኪ
ሱ
ሶ
ውኩ
ኩ
ካይ
ዳይ
ስታኒ
ኑኒ
ሻፅኾ
ስኾ
ትኾ
ትሻ
ሻ
ፁኻ
ሜቴን
ጃን
ናን
ችን
ን
ዋላ
ካሊ
ቴንካው
ካላው
ሻፃው
ሚው
ኩው
ስቱንኩ
ስታንኩ
ሻፅናኩ
ችንኩ
ኒ
ሻቄና
ሻፃና
ስታና
ታና
ና
ፅኾ
ንኾ
ኾ
ፅሻ
ሻኾኻ
ውላ
ኪላ
ጃዊያኽ
ስታዌኽ
ካማኽ
ያኽ
ካው
ታው
ጋው
ፂው
ኒው
ው
ቱንኩ
ጃንኩ
ካኩ
ስኩ
ሻኾኹ
ስትኹ
ቱንኻ
ችኹ
ንኹ
ኹ
ኻንኾ
ስትሻ
ንሻ
ኾኻ

Appendix II: compiled Rule sets

Rule set 1

```
if word ends with('ስ')  {
     if (word ends with('ታንኻስ ') or ('ኒዴስ') or ('ካዴስ') or ('ውዴስ') or ('ፅኻስ') ('ትንስ') or ('ኻፀስ') or
    ('ፅንስ')or  ('ኻኺስ')or ('ስኻስ'))  {
              remove the matching suffix
     else if (word ends with('ኻስ') or ('ትስ') or ('ዴስ') or ('ዳስ') or ('ንስ') or ('ካስ') or ('ኑስ')  or('ኺስ')) {
              remove the matching suffix
     else
               remove the suffix
              }
         }
     }
```

Rule set 2

```
else if word ends with('ካ' ) {
     if (word ends with('ጃንትካ') or ('ስቲካ') or ('ኻፅካ')){
            remove the matching suffix
     else if (word ends with('ፅካ') or ('ውካ')) {
        if word length>3
           remove the matching suffix
  else:
              remove the suffix 'ካ'
              }
       }
  }
```

Rule set 3

```
      else if words ends with('ታ')  {
              if (word  ends with('ካዋስታ') or ('ኩሳስታ') or  ('ንኹስታ') or ('ካዳስታ') or('ካውስታ')or
             ('ዴስስታ')or ('ካስስታ')) {
                       remove the matching suffix
              else if (word ends with('ካስታ') or ('ዋስታ') or ('ሚስታ') or ('ስስታ') or ('ዳስታ') or ('ውስታ') or
             ('ኻስታ') or ('ኻፂታ') or ("ኪስታ')){
                          if word length>5
                                    remove the matching suffix
              else if (word ends with('ስታ') or ('ኻታ') {
```

remove the matching suffix
else
if word length>3
remove the suffix ('ታ')
}
}
}
}

Rule set 4

else if word ends with('ሳ') {
if (word ends with('ችንኩሳ') or ('ታንኩሳ') or ('ሻፅኹሳ')) {
remove the matching suffix
else if (word ends with('ካውሳ') or ('ውኩሳ') or ('ሚውሳ') or ('ናውሳ')) {
remove the matching suffix
else if (word ends with('ኩሳ') or ('ውሳ') or ('ኽሳ') or ('ችሳ')or ('ኹሳ') or ('ኑሳ')) {
remove the matching suffix
else:
remove the suffix('ሳ','')
}
}
}
}

Rule set 5

else if word ends with('ቲ') {
if word ends with('ላንቲ') {
replace the suffix('ላንቲ') with ('ል')
else if (word ends with('ስታንቲ')) {
if word length>2
remove the matching suffix
else if (word ends with('ዋንቲ') or ('ፃንቲ') or ('ቻንቲ')) {
if word length>2
remove the matching suffix
else if word ends with('ንቲ') {
if word length>
remove the matching suffix
else
if word length>2
remove the suffix ('ቲ')

```
            }
        }
    }
```

Rule set 6

```
else if word ends with('ን') {
        if (word ends with('ስቱን') or ('ሜቴን')) {
                if word length>3
                    remove the matching suffix
        else if (word ends with('ጀን') or('ናን') ('ችን')) {
                if word length>3
                     remove the matching suffix
        else
                if word length>3
                     remove the suffix  ('ን')
                }
            }
        }
```

Rule set 7

```
else if word ends with('ኽ') {
        if (word ends with('ጀዊያኽ') or ('ስታቄኽ') or ('ካማኽ')) {
                remove the matching suffix
        else if (word ends with('ያኽ') or ('ችኽ') or ('ኼኽ') or ('ቄኽ') or ('ዳኽ')) {
                remove the matching suffix
        else
                if word length>2
                    remove the suffix   ('ኽ')
            }
        }
    }
```

Rule set 8

```
else if word ends with('ው')  {
        if(word ends with('ቴንካው') or ('ካላው')  or ('ሻፃው')) {
                remove the matching suffix
        else if(word ends with('ሚው') or ('ኩው') or('ካው') or ('ታው') or ('ጋው') or ('ዚው') or ('ኒው')) {
                remove the matching suffix
        else
                if word length>2
                        remove the suffix ('ው' )
            }
```

```
        }
    }
```

Rule set 9

```
else if word ends with('ኩ') {
        if  (word ends with('ስቱንኩ') or ('ስታንኩ') or ('ችፅናኩ')) {
                remove the matching suffix
        else if (word ends with('ችንኩ') or ('ቱንኩ') or ('ቻንኩ')) {
                remove the matching suffix
        else if (word ends with('ካኩ') or ('ስኩ') or ('ውኩ'))  {
                remove the matching suffix
        else
                remove the suffix   ('ኩ')
                }
            }
        }
 }
```

Rule set 10

```
else if word ends with('ና')
        if (word ends with('ችቴና') or ('ችፃና') or ('ስታና'))
                remove the matching suffix
        else if word ends with('ታና')
                remove the matching suffix
        else
                remove the suffix ('ና')
```

Rule set 11

```
else if word ends with('ኹ') {
    if (word ends with('ችኾኹ') or ('ስትኹ') or ('ቱንኻ'))  {
            remove the matching suffix
    else if (word ends with('ችኹ') or ('ንኹ'))   {
            remove the matching suffix
    else
            if word length>2
                    remove the suffix  ('ኹ','')
        }
      }
}
```

Rule set 12

```
else if word ends with('ኹ') {
        if (word ends with('ሻፅኹ') or ('ፅኹ') or ('ኻንኹ')) {
                if word length>3
                        remove the matching suffix
        else if (word ends with('ስኹ') or word ends with ('ትኹ') or word ends with('ንኹ'))  {
                remove the matching suffix
        else
                remove the suffix   ('ኹ')
                }
           }
      }
```

Rule set 13

```
else if word ends with('ሻ') {
        if (word ends with('ስትሻ') or ('ትሻ') or ('ፅሻ') or ('ንሻ')) {
                remove the matching suffix
        else
                remove the suffix   ('ሻ')
                }
            }
       }
```

Rule set 14

```
else if word ends with('ኻ') {
        if (word ends with('ሻኹኻ') or ('ኹኻ') or ('ፁኻ')) {
                remove the matching suffix
        else
                if word length>2
                        remove the suffix('ኻ')
           }
      }
```

Appendix III: Sample Test set Data

ይወትስ ዝኩኑስ እኑስ እኑዋ ካሲት ሻናው፣ እምኘልዳ ኹናው ዝቕናውሳ ዴሶ እሜማ ማቤራዊ ኬሴትስ ታንባንኩሳ ዲብካዋ ኽይናው ኽሳንቲ እንፃንቴ እምኘል ዳድስ እንፂውሻፃው ኮንኪስ፣ አሳብስ አስሜምሻፃው እምኘል ቤረሰቡ አይቱ ጊሊፂ ያኻማ አግስቴ፡፡ አዊ እዝብ ኚጆርትጆሱ ዴሶ፣ ባይልስታ ቺፅሻፂው አኺኒ ዝኩኽ ያኹስ አዊ ላኼታዳ ኪላ እንፃንትካስታ ኽሳንትካ ያኹንኩ ሊጊሲሚ ጊዝኩ ታሪክዌና ባይልካ ዙራሙሪ ቺፋ አግስታና፡፡ ዙራሙሪ ቺፋ ቲሪትጆንኩ ማቤርካ፣ ድርካ እንኚታኪ ሊሊትኾንኩ እምኘልታቕኩ ታምትሻፅካ ዳድካ አኹኪ ባይልካ እምኘል አሴብንኹ ዲብስ ይባቺስ ያኻውላ ሚንቻ ያኹንኩሳ ቴግባርካዋ አኮሜቻኒስ ካንትስታና፡፡ እኒጆ ክቻዴስ ማቢሬ ካንትኑኒ ባይሎ ኬቤርፅሻስ ፂፂንኩ ማቤርካ ሚንችካ ቴግባርካስ እሼና፡፡ አሬክሻፅካ፣ ልማትኩ ኩስሻፅካ፡ ዛኽ ዝሜድሊ ካሲትሻፅካ ያኻኒስ ካንትስታና፡፡ እኒ ዙራሙሪ ቺፋ ቲሪትሻፅናኩ ባይልኩ ታምትሻፅካ እሼኑ ቴግባር ኪላ እን ዙራኽ እስታው ጋቲዊያኽ፡፡ አዊ ዞንሾ ጌምሻስ ባንጂ ዋረዳዴስ ፌይሻ እናና ጊምቢኼ ጌምናኒ ቴርቴራ ሹኻ ብሽታንካ ካንትስታና፡፡ እኖጅኪላ ዋሺኒ ብሽታንዳ ሹማጌልካዋ አግፅናው አኽጐህ ታክስሻስ ካኖ ኬምኘ ኬንፓስፍናና ዚቾ ጐሜርቲ እስታው ዙራሙሪሾ እሸኹዋ ጉሪኚስ ጁናና ጋሳጋስንኹ፡፡ እሌት አድጉዊ ማንዚ 15/32ዐዐ4 ምሬት አሜት ሲዚ ሳትዳ ብሽታና ታምናና ፉሪሻ እንኾ እንኚ እንዝኚስ ኪላ እንዳራ አግፂን? ናውሳ ካሴ ጉሸታንኻስ እንቶጂስ ዙርሚ ያኻ እንኹዋሳ እንታ ኻናና ዲና ክርፂ ንባብ!

1948 ምሬት አሜታዴስ ጇሜራማ ላንቡሳ ባይሎኮንቖስታማ አይቱሳ ዝኩዊኔእንፂውሻፅኹሳ ማቢሬ ቲሪትሻፅካማ እምኘል ክራኒ እምኘል ቲኪስትሻስ እስካውካ ካሜንስታንትካዋ ቲካማጊ ላንቡሳ ባይል ማንዲስታማ አግስቴ፡፡ ማቢሬ ካንትኑስ “ኽታዊኒ ጊዝዳ አዛ ቻካጃ ናንቲ ባግፃ እስቴት ኹና ስልኺታኪ እኾሼ ኩርስታኪ ግምሴ ዜኬርሻስ ቲሪትሻፂኹሳ ባለግዛቤሩሳ ማቢሬ ባናላ እናና ላንቡሳ እምናና ታሪኮ ማንዲሻስ ዝኩና ናኒስ አስቖኒ አርባ ቴንሳዳ ዝኩዋንትካ ያኹንኹ ብላታ ዘለከ እንየው እንክርፅካ፡፡ እን ማቢሪዳ አባልካስታ ሹማጌልካ አኽሻስ ቺፅሻፃኒስ ታብሊ ታብላሱሳ ማቢሬ ድፅናላ ይንታው ክምንትስ ኪላ ፌያፌይሻፅናኻ ናኒስ ባለግዛቤሩ ማቢሪው አባልካሊ አኽሻስ እንታ ዙሚትኼ ፄውና፡፡ ብላታ ዘለከ እንየው ዙሚታጊ ባግሳ ናኑስ እስታው ኼራሳ ሚፄታ ዚቾ ጐሚርቲ እስታው ስፍርሾ አንትሻስ ጀርካዋ ካሜንታታ ሴሻቱስ እንሳ ባለግዚአቤሩሳ ማቢሬ ቲሪትሻፄታ ጀርጀራስጊ አዴሮ ይታታ ክታ፡፡ እንሳ አዴሮ ይቱስ ኪላ “እንሳ ማቢሬ ፍሸቱንኻሳ ፍሸምባ ድፂንኻሳኩፂ ድፅምባ” ንሻስ ካሎ አዴርስ ይታታ እን አሌምዴስ አጉሌልሻታ፡፡ እንሳ ካሎ ቻቤልሻስ እኖጂኪላ እንሳ ማቢሬ ታብሊ ታብላዴስ ቻቤልናና አኒኩ ላንቡሳኺስታ እንሳላካው ዝቤንስ ማንዲናና አኮሜችናናጊ አግስትኔ ናኒስ ኽይትሻፅካ፡፡ ፄውንኹ እጂኚዳ ኪላ እን ማቢሪው ቹዋ ቴግባር እንዳር አኽጐህ ኪላ ካሊጐህ ካስኑካ፡፡ ኃ እዩኑ ዙርሚኪላ እን ማቢሪው ቹዋ አላሚ ማቢሬ ዝቕሻ ይባቺ ያኻውላ እን ማቢሪው ስምስ ዝኩኹሳጊ አቕ ክራኒ ዴቄካማ ኮሬብፅካማ፣ ቼጌርስታኒ እርዳትካማ ጌውስጀኒኪ አሬክሻፅካማ፣ እምኘልታቕ ባይልስ እንፄውሻፅካማ እንካንስ ዝኩዋንታ ፄውጐህ ቴግባር ዝኩኽ ማቢሪ አኽጐህ ጌሌፅካ፡፡ እንሳ ማቢሬ አችትሻፃኒስ ኪላ አጋፋሬ ሊክሙሳ እንኚታኪ ፃፌ ዴሜክካማኽ፡፡ ኽሳንቲ ማቤሬ ፄውካማ እንሳ ማቢሬ ዝቓው እምኘል ዙራሙሪው አቕ ይባቺ ያኻውላ ውላ አዊ ላኼታዳ አግስታው፣ ቻሪ፣ ባንጂ፣ አንኪሼ ዴሜካማ ውላዋ አሴቱፍሻፅኹዊያኽ፡፡ እን ማቢሪው ኽሳንቲ ቴግባር ኪላ አጌር ቺፋ ጌውስሻኹሳ፣ እንኬ ድስሻኹስ ያኹኒ ይንታማ አሬክጀው እን እስቕኒ እርባቴንሳው ባለግዚአቤሩ ማቢሪዳኽ፡፡