ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
SCHOOL OF INFORMATION SCIENCE

DEVELOPMENT OF STEMMING ALGORITHM FOR TIGRIGNA TEXT

YONAS FISSEHA

JUNE 2011

ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
SCHOOL OF INFORMATION SCIENCE

DEVELOPMENT OF STEMMING ALGORITHM FOR TIGRIGNA TEXT

By

YONAS FISSEHA

JUNE 2011

ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
SCHOOL OF INFORMATION SCIENCE

DEVELOPMENT OF STEMMING ALGORITHM FOR TIGRIGNA TEXT

By

YONAS FISSEHA

Name and signature of Members of the Examining Board

Name	**Title**	**Signature**	**Date**
_____	Chairperson	_____	_____
_____	Advisor(s),	_____	_____
_____	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.

_____

Date

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

_____

Advisor

A C K N O W L E D G E M E N T S

Here, I would like to take this opportunity to express my most gratitude to those that had helped and inspired me in the completion of this thesis. I owe a great debt of gratitude to my advisor Ato Ermias Abebe. This thesis would not have been possible without his guidance, as well as inspiring and enlightening ideas, comments and suggestions. I am glad to be his student and share his rich, broad and deep knowledge. Thank you very much.

To my friends, thanks for all the moral supports that they had given me and many other individuals that are unable to be listed who have directly or indirectly help me in completion of my thesis.

I would like to express my gratitude to my beloved family for their encouragement, patience, and financial support they have given me during the course of this thesis.

Last but not least, I would like to thank the University of Addis Ababa for the financial support without which the study would not have been possible.

Yonas Fisseha Gidey
June, 2011

TABLE OF CONTENTS

List of tables

ABSTRACT

This paper presents the development of a rule-based stemming algorithm for Tigrigna. The algorithm is simple yet highly effective; it is based on a set of steps composed by a collection of rules. Each rule specifies the affixes to be removed; the minimum length allowed for the stem and a list of exceptions rules. In Tigrigna language there are many exceptions for making any stemming rule. The researcher has considered these exceptions in designing the stemmer.

The deep study of the Tigrigna grammar as well as the analysis of the inflectional and derivational types of affixes of the language was necessary for this kind of thesis work. The stemmer was designed by new word classification according to their affixes. The stemming is performed using a rule-based algorithm that removes affixes.

Research done for Tigrigna language and Tigrigna stemmer was taken in to consideration. It was necessary to conduct the research as the past research of Tigrigna language stemming is limited. By Analyzing the Tigrigna grammatical rules, the researcher decided to follow inflectional and derivational affix removal and designed a new rule-set for the Tigrigna stemmer.

The goal of the research was to develop and document a new rule-based stemmer for the Tigrigna language. The Tigrigna stemmer was developed in Python programming language. The researcher tried to follow a simple structure in the algorithm, creating

small rule-sets for similar affixes, which are working as Rule-sets on the input words.

The stemmer was evaluated using error counting method. The system was tested and evaluated based on the counting of actual understemming and overstemming errors using a total of 5437 word variants derived from two data sets. Results show that the stemmer has 85.8 % accuracy for the first dataset and 86.3% accuracy for the second dataset and average accuracy of 86.1%. The proposed method generates some errors. The average error rate is about 13.9%.These errors were analyzed and classified into two different categories (overstemming and understemming). Most of the errors occurred due to overstemming of words.

CHAPTER ONE

INTRODUCTION

1.1 BACKGROUND OF THE STUDY

Stemming is a procedure, which reduces all words with the same root to a common form by stripping from each word its derivational and inflectional suffixes (Lovins, 1968). Popovic & Willett (1992) also described it as one of the well-known methods that are used to identify morphological variants. Common to all languages, words variants are formed by the usage of affixes, alternative spelling (that is different spelling for single word), transliteration, abbreviations and spelling errors. The need for stemming arises from the fact that natural languages are characterized by morphological variations of words that might have several forms resulting from the addition of different affixes. The main objective of stemming algorithms is to remove all possible affixes and thus reduce the word to its stem (Dawson, 1974).

Historically, most of the researches on stemming were motivated by information retrieval. Nowadays Stemming has been widely used in several fields of natural language processing such as, information retrieval, data mining, text classification and categorization, text compression, data encryption, spelling aids and automatic machine translation (Lovins, 1968).

In information retrieval, the relationship between a query and a document is determined primarily by the frequency of terms, which they have in common (Hull D.A, 1996). However, some words may have a number of morphological variants that need some form of natural language processing to be recognized. As information retrieval is becoming more and more sophisticated, trying to cover user's need to access specific information, it is necessary to make IR systems more effective using some mechanisms. One of the attempts to make information retrieval more effective in retrieving documents that satisfies user's specific demand is the usage of stemming. The vocabulary mismatch

problem characterizes the main information retrieval problem. The problem occurs when the form of a word in a query does not match the forms found in documents relevant to the query.

Stemming is one of many tools used in information retrieval to resolve the vocabulary mismatch problem, in which user query words do not match document words. For example, if a user uses the Tigrigna query teSewatay ("ተፀዋታይ" ,the player) some inflectional forms of the word "teSewatay" exist by attaching many letters to the Tigrigna root word *Sewata(ፀወታ, play).* Thus, a user's query that contains the tigrigna word *teSewatay (ተፀዋታይ, player)* will not match any documents that contain the following Tigrigna words: teSawati (ተፀዋቲ, a player*), etiteSawati (እቲተፀዋቲ, the player), kemtiteSawati(ከምቲተፀዋቲ, like the player), netiteSawati (ኑቲተፀዋቲ, for the player), btiteSawati(ብቲተፀዋቲ, by the player), teSawatin (ተፀዋቲን, and player).* All documents that contain the other variant forms of the word teSewatay ("ተፀዋታይ", the player) would not be retrieved, because there is no matching between words in the user's query and these documents. It is clear that the high inflection of Tigrigna language can result in the problem of vocabulary mismatch, which in turn, reduces the accuracy of information retrieval significantly. The process of information retrieval will be improved greatly when stemming is used to avoid the vocabulary mismatch problem.

In automated morphological analysis, stemming is also employed to derive the root word and its affixes from a given word and the relationship between the root word and its affixes. The affixes can be used as guiding clues to grammatical structure. Thus, a stemming algorithm is required to recognize the morphological properties of these affixes (Hui.B., 1998).

In Automatic document classification and categorization, stemming is applied to stem morphologically related variants of the same word so that it can help to classify similar documents based on their contents. With the growth of text documents, information retrieval has become a crucial task to satisfy the needs of different end users. To this end, automatic text categorization and classification has emerged as a way to cope with such a

problem. Automatic document classification attempts to replace and save human effort required in performing manual classification. The importance of the stemming process in the classification is to make the operations of classification less dependent on particular forms of words and reduces the potential size of vocabularies, which might otherwise have to contain all possible forms (Almuhareb, 2006).

According to Frakes (1992) the main advantage of stemming algorithms particularly for IR purposes is improving IR performance to provide searchers with ways of finding morphological variants of search terms. Stemming is also used in IR to reduce the size of index files. Since a single stem typically corresponds to several full terms, by storing stems instead of terms, compression factors of over 50 percent can be achieved. Reducing the dictionary size by stemming terms can be high in various NLP applications, especially for highly inflected languages.

Various stemming algorithms for a number of languages have been proposed. In Most cases the design of stemmers is language specific, and requires some significant linguistic expertise in the language, as well as the understanding of the needs of writing system for that language (C.J. van Rijsbergen, 1980).The structure of these stemmers range from the simplest technique that involves removing suffixes using a list of frequent suffixes, while a more complex one would use morphological analysis of the language to derive a stem from the words. Consequently, a stemmer's performance and effectiveness in NLP applications vary across languages.

1.2 STATEMENT OF THE PROBLEM

Each natural language has its own characteristics and features. So, it seems 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 English textual documents.

However, little attention has been given to stemming of textual data in other languages, especially to Tigrigna language.

Tigrigna is the native language for the Tigray and Eritrean people. It is one of the widely spoken languages in Ethiopia and Eritrea. It is also spoken in some parts of the world especially in the Middle East, Europe and North America by Ethiopian and Eritrean immigrants living in these areas. Tigrigna is morphologically very complex and a highly inflected language. It uses both kinds of morphologies, i.e. inflectional and derivational (Kassa G., 2004). Both inflectional and derivational morphologies in Tigrigna result in very large numbers of variants for a single word. Tigrigna clearly belongs to the set of severely under-resourced languages and there has been almost no computational work on the language. For a language with the morphological complexity of Tigrigna, a crucial early step in IR and NLP work must be the development of stemming algorithm. Consequently Tigrigna requires superior stemming algorithms for effective information retrieval and natural language applications.

As Tigrigna is morphologically complex language, there is a need for automated procedures that can reduce the size of a word list to a manageable level and improve retrieval performance, and also capture the strong relationships existing between different word forms in the language. Stemming plays an important role in the identification of a word stem from a full word by removing inflectional and derivational affixes, and there has been much interest for developing algorithm for this purpose. This interest is likely to increase still further as more and more types of text-processing applications become of wide spread importance.

There are lots of digital documents in Tigrigna language available on the Internet and the World Wide Web which are accessible for users. In order to provide the necessary access to this wealth of information and to enable its development, the basic information retrieval tools need to be designed and deployed. Developing stemming algorithm for Tigrigna language can also pave a way to develop other natural language processing such

as, text classification, text summarization, machine translation, text categorization and morphological analyzer.

Girma (2001) applied stemming algorithm for Tigrigna texts . This algorithm adopted techniques from the Porter stemmer (Porter, 1980) for describing the rules used by the stemmer. The performance of Girma's algorithm was tested by using textual data taken from few resources and with small size of the sample texts in terms of number of unique words.This algorithm can't be considered as rule-based stemmer even if the algorithm has set of rules that simply strip the affixes. Adoption of the Porter stemming algorithm for highly inflectional languages like Tigrigna is not recommended because the Porter algorithm is appropriate only for languages which make word variants through suffixation (Larkey, 2002). In addition, it would have shown conveying results had it been tried on text collection of large size collected from different sources.

Girma (2001) recommended that specific grammatical rules can improve the effectiveness of the stemming algorithm for Tigrigna texts. Therefore, the researcher has taken this opportunity to develop a new rule-based stemmer to improve the effectiveness of the stemming of Tigrigna text. The development of a Tigrigna stemmer with extended grammatical rules will hopefully come to solve the problem of the previous algorithm.

1.3 OBJECTIVES

1.3.1 GENERAL OBJECTIVE

The main objective of this research is to develop rule-based stemming algorithm for Tigrigna language text.

1.3.2 SPECIFIC OBJECTIVES

The specific objectives of the research are:

- To review the morphology of Tigrigna and stemming algorithms
- To review properties of the Tigrigna text in order to get familiar with the different aspects of the language
- To compile a list of affixes used in Tigrigna text
- To compile functional words of Tigrigna
- To Develop stemming algorithm(s) to experiment with Tigrigna text
- To write computer program for stemming affixes and removing stopwords

1.4. METHODOLOGY

1.4.1 REVIEW OF RELATED LITRATURE

Since stemming in most cases is language dependent, to develop stemming algorithm one has to know the morphology of the language or work with a linguistic expert. To understand the morphology of Tigrigna, review of works on the language is done by consulting different sources such as books, textbooks, journals etc. Additional information is collected through personal discussion with professional experts of the language.

There are a number of stemming algorithms developed for different languages. The approaches and techniques used in these algorithms especially for English and Semitic languages will be studied from different resources.

1.4.2 DATA RESOURCES

A text corpus is one of the resources required in natural language processing researches. A good sized text can reasonably show a language's morphological behavior. Selection of text is, therefore, an important component in developing a stemmer. For the purpose of this research, the researcher uses a corpus which can be representative of the language. A sample text of different disciplines is collected from different sources.

1.4.3 PROGRAMMING TECHNIQUES

A program was developed using the Python programming language to implement the stemmer. This is because Python has rich string manipulation techniques and the researcher has some experience of writing programs using Python.

1.4.4 DEVELOPING AND TESTING THE ALGORITHM

Tigrigna language has rich morphology, the process of stemming involves dealing with prefixes, and derivatives in addition to suffix stripping. The algorithm is developed by analyzing morphological rules of the language. The experimentation of the stemmer is done on text collection of relatively large size collected from different sources. The Algorithm is tested using error counting technique. The extent of accuracy of the stemmer is determined by qualitative analysis of the output from the stemmer.

1.5 SCOPE AND LIMITATIONS OF THE STUDY

The researcher has introduced this algorithm in order to cover the gap of the existing Tigrigna algorithm and create a more effective Tigrigna stemmer. Based on the previous research of Girma (2001), the researcher created a system that removes the affixes of the Tigrigna words. An accurate Tigrigna stemmer can be used for various purposes in information retrieval and morphological analysis. In this research the algorithm handles only prefixes and suffixes. Reduplication, compounding and irregular words are not handled in this research due the irregularities in the affixation of these words.

1.6 ORGANIZATION OF THE THESIS

The thesis is presented in five chapters. The first chapter presents an explanation about the research background, problem description, objective of the study, methodology as well as scope and limitation of the study.

Chapter two presents the various concepts in stemming and the application of stemming algorithms in various languages. Several types of stemming algorithms for languages such as Tigrigna, Amharic, Wolayta, Oromo, English and Arabic are discussed.

Chapter three discusses the morphology of Tigrigna in general. It also describes the overview of Tigrigna language, word formation in Tigrigna and derivational and inflectional morphology of verbs, nouns, and adjectives.

Chapter four presents the design and implementation of the stemmer. The design of the stemmer discuses about the document collection used in the research, the morphological preprocessing, construction of stopwords, compilation of the affixes and the proposed Algorithm. The findings gained from the analysis of the result of stemming algorithm are discussed in this chapter.

Finally, Chapter five presents summary, conclusion and recommendation.

CHAPTER TWO

LITRATURE REVIEW

2.1 INTRODUCTION

One of the first problems related to the use of natural language in information retrieval and Natural Language Processing applications is that of morphological variation of words. Morphological variation of words refers to the fact that words may occur in inflected forms, or that derivation is used to produce new but related words, or words are combined into compound words. In most cases, morphological variants of words have similar semantic interpretations and can be considered as the same for the purpose of IR and NLP applications.

In natural language, the number of possible inflected and derived form is very high, and the variation of words is due to the attachment of prefixes, infixes and suffixes to a word. Stemming i.e. the removal of prefixes infixes and suffixes from words using different possible techniques, is therefore considered a practical way to match related word forms to a common stem. Terms with common stems tend to have similar meaning, which makes stemming highly important for retrieval and NLP applications. M.Popovic and P. Willet in their research described that Morphology as the main sources of word variation in free-text retrieval systems (M.Popovic and P. Willet 1992). The most common way of dealing with this type of variation is by means of a stemming algorithm, which reduces all words with the same root to a single form by stripping the root of its derivational and inflectional affixes.

Stemming is a technique used to conflate or reduce morphological variants of words to a single term called stem. According to Lovins (1968) stemming algorithm “is a computational procedure, which reduces all words with the same root to a common form by striping each word of its derivational and inflectional affixes”. Stemming also includes the striping of prefixes, infixes and other attachments to a word. Stemming algorithms are very important because they increase the efficiency of document retrieving systems

and reduce the size of index files due to their grouping of many morphological term variants into one single stem. Stemming is language-dependent and should be customized for each language, since languages have a varying degree of differences in their morphological properties.

The main objective of the stemming process is to remove all possible affixes and thus reduce the word to its stem (Dawson 1974). Search engines use stemming, to match words with prefixes and suffixes to their word stem and this makes the search to include many options in the meaning that it can ensure that the maximum number of relevant documents is included in search results. Stemming also has applications in machine translation, document summarization, text categorization, thesaurus construction, spelling checkers and text classification.

For IR purposes, it doesn't usually matter whether the stems generated are genuine words or not. The key terms of a query or document are represented by stems rather than by the original words. This not only means that different variants of a term can be conflated to a single representative form – it also reduces the dictionary size, that is, the number of distinct terms needed for representing a set of documents. A smaller dictionary size results in a saving of storage space and processing time. Frakes & Baeza-Yates(1992) in their paper addressed stemming as the basic text processing tool often used for efficient and effective text retrieval and it is one of many tools used in information retrieval to combat the vocabulary mismatch problem, in which query words do not match document words.

2.2 STEMMING ALGORITHMS

Stemming is the process for reducing inflected, or derived words to their stem, base or root form . The stem need not be identical to the morphological root of the word. It is usually sufficient that related words map to the same stem, even if this stem is not in itself a valid root. Stemming algorithm has been addressed as problem in the areas of information retrieval and NLP long time ago and the first paper on the subject was published in 1968(Lovins 1968). The process of stemming, often called conflation, is

useful in search engines for query expansion or indexing and other natural language processing problems. In information retrieval, the relationship between a query and a document is determined primarily by the number and frequency of terms which they have in common. Unfortunately, words have many morphological variants which will not be recognized by term-matching algorithms without some form of natural language processing. In most cases, these variants have similar semantic interpretations and can be treated as equivalent for information retrieval (as opposed to linguistic) applications. Therefore, a number of stemming or conflation algorithms have been developed for IR in order to reduce morphological variants to their root form. Stemming programs are commonly referred to as stemming algorithms or stemmers.

One of the key questions in stemming algorithms is if a computerized stemming can be as much efficient as a manual processing for IR purposes. Another important and related problem is if a word should be reduced only on the right root morpheme boundary or at a non-linguistically correct point (Frakes, 1984). Despite that stemming often relies on knowledge of language morphology, it does not mean its goal is to find a proper meaningful root of a word. Instead, a word can be striped at a position “incorrect" from the natural language point of view. For example, the M.F. Porter's algorithm (Porter 1981) can make incorrect productions of words in which the results are not morphologically right forms of words. Nevertheless, since document index and queries are stemmed "invisibly" for a user, this particularity should not be considered as a flaw.The problem of stemming has been approached with a wide variety of different methods. According to the Frakes’ classification (1992), all stemming algorithms can be roughly classified as Dictionary-Based Technique, Successor variety, Affix removal and Statistical (N-gram).

2.2.1 DICTIONARY-BASED TECHNIQUE

Frakes & Baeza-Yates (1992) defined dictionary-based technique as one way to do stemming by storing a table of all index terms and their stems. A dictionary technique depends mainly on creating a very large dictionary which stores words found in natural texts with their corresponding morphological parts. Such parts include: stems, roots, and affixations. Each word uses a unique entry in a lookup table. For example terms such as engineering, engineered, engineer are stemmed to engineer. Under this approach, the stemming process is performed manually, wherein the stems are defined for each word and stored in some kind of structured form.

This technique lists all words that exist in a specific language with their corresponding variation of words created by different attachments to the root word. To stem a word, the table is queried to find a matching variation of a word. If a matching variation of a word is found, the associated root form is returned. Terms from queries and indexes could then be stemmed via table lookup. These dictionaries are usually constructed manually. In this technique, words could be stemmed via a table lookup. Table entries are listed in an alphabetical order. A hash table or binary search list can be used to optimize the search. Even though such approaches are accurate, the problems associated with them include the dictionary maintenance, to keep up with an ever-changing language, and this is actually quite a problem. It is not only that a dictionary created to assist stemming nowadays will probably require major updating in a few years time, but also that a dictionary in use for this purpose today may already be several years out of date as well as storage overhead and retrieval time needed for such data.

The advantage of this approach is that it generates perfect stems. However, the approach is limited to retrieving only those words that have previously been stored. The space occupied for storage tends to grow as the corpus expands, which can make the search process inefficient. In Krovetz's dictionary experiments (Krovetz 1995), this direct method, seems effective but inadequate to deal with the "unlimited" words and their formation, especially in inflected languages with elevated morphological structure. This

was the main reason that led him to evaluate algorithmic stemmers and conclude that "despite the errors they can be seen to make, they still give good practical results". It is not only that a dictionary created to assist stemming nowadays will probably require major updating in a few years time, but also that a dictionary in use for this purpose today may already be several years out of date".

2.2.2. SUCCESSOR VARIETY

The successor variety approach has been introduced by Hafer and Weiss. It is corpus-based and observes the number of distinct letters following a particular prefix: the successor variety. Successor variety stemmers (Hafer and Weiss 1974) are based on work in structural linguistics which attempted to determine word and morpheme boundaries based on the distribution of phonemes in a large body of words. The stemming method based on this work uses letters in place of phonemes, and a body of text in place of phonemically transcribed words. It scans the word to be stemmed and finds the cut point where the successor variety increases sharply. Several variations are possible, including: cut-off, peak and plateau, complete word and entropy.

2.2.3. AFFIX REMOVAL ALGORITHMS

This approach is dependent on the morphology of the target language. Affix removal algorithms remove suffixes and/or prefixes from terms leaving a stem. These algorithms sometimes transform the resultant stem. There are many varieties of affix removal algorithms. Most stemmers currently in use are iterative longest match stemmers, a kind of affix removal stemmer first developed by Lovins (1968). An iterative longest match stemmer removes the longest possible string of characters from a word according to a set of rules. This process is repeated until no more characters can be removed. Even after all characters have been removed, stems may not be correctly conflated. In addition to Lovins(1968), iterative longest match stemmers have been reported by Dawson (1974), Porter (1980), and Paice & Husk (1990).

Affix removal algorithms are often called Rule-Based Technique and is a widely applied stemming technique and the algorithm is introduced by Porter (1980). With specific rules for the English language, this algorithm iteratively removes suffixes from a given word, reducing it to its stem. Even if the algorithm has its limitations, it is the most commonly accepted for its high precision and recall. Lovin's stemmer (1968) follows the same rule-based technique but it does not apply its rules iteratively and it is more conservative than Porter's algorithm.

2.2.4. STATISTICAL APPROACH

The stemming process involves statistical methods whereby, through a process of inference and based on a corpus, rules are formulated regarding word formation. Some of the methodologies adopted are N-gram (Mayfield J., 2003) and Hidden Markov Models (HMM) (Melucci, M.2003).Most stemmers are language-specific. Single N-gram stemming suggested by James Mayfield and Paul McNamee (2003) tries to bypass this limitation. The idea is to analyze distribution of all N-grams in a document. Since the morphological invariants (unique word roots) will occur less frequently than variant parts (common prefixes and suffixes), a typical statistics like inverse document frequency (IDF) is used to identify them. The authors successfully tested the algorithm on some European languages. The N-gram stemming underperformed stems and required significant amount of memory and storage for index, but its ability to work with an arbitrary language makes it useful for many applications.

Another statistical approach to stemmers design used by Massimo Melucci and Nicola Orio (2003) is to build a stemming algorithm based on Hidden Markov Models (HMM). It doesn't need a prior linguistic knowledge or a manually created training set. Instead it uses unsupervised training which can be performed at indexing time.HMMs are finite-state automata with transitions defined by probability functions. Each character comprising a word is considered as a state. The authors divided all possible states into two groups (roots and suffixes) and two categories: initial (which can be roots only) and final (roots or suffixes). Transitions between states define word building process. For any given word, the most probable path from initial to final states will produce the split point

(a transition from roots to suffixes). Then the sequence of characters before this point can be considered as a stem. Using Porter's algorithm as a baseline, they found that HMM had a tendency to overstem the words.

In their research, Jinxi Xu and W. Bruce Croft suggested an approach, which allows correcting "rude" stemming results based on the statistical properties of a corpus used (Xu J., Croft B 1998). The basic idea is to generate equivalence classes for words with a classical stemmer and then "separate back" some conflated words based on their co-occurrence in the corpora. This approach does not require any linguistic knowledge whatsoever, being totally independent of the morphological structure of the target language.

2.3 STEMMING ALGORITHMS FOR ETHIOPIAN LANGUAGES

Unlike English and other western languages, Ethiopian languages are less researched languages in the areas of information retrieval and natural languages processing applications. Recently there are some researches done in the areas of IR and NLP for Ethiopian languages. Some of the attempts done are presented as follows.

2.3.1 TIGRIGNA STEMMERS

The first attempt to develop Tigrigna stemmer was made by Girma Berhe(Girma Berhe,2001). The algorithm uses iterative approach but when it finds two affixes that match with the word, it removes the longest one. As Tigrigna is morphologically complex language a context-sensitive stemmer was considered appropriate in this algorithm. Each affix is accompanied by a context-sensitive rule. The techniques for describing the rules are adopted from Porter (1980). 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.

This stemmer used 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. This step takes 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. In removing a prefix, checking for match in the prefix list and counting length of the remaining string is done. 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.

2.3.1 AMHARIC STEMMERS

Amharic and Tigrigna are both Semitic languages and have similar morphological system. The inflection and derivation of words follows similar pattern. So, reviewing researches of stemming algorithms done for Amharic language helps for developing Tigrigna.

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 (2002, 2003) which investigated the effect of stemming for information retrieval. Nega Alemayehu and Peter Willett (2002) have developed a stemmer for information retrieval purposes. The stemmer has been developed in a manner analogous to that used previously in a stemmer for Slovene (Popovic and Willett, 1990). Specifically the algorithm first identifies a set of stop-words and then a set of affixes associated with the remaining content-bearing words. They have used the characteristics of the resulting affixes were used to guide the development of the stemmer. 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 andWillett (2003) 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 (Argaw and Asker 2007). 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 a limited text consisting of 50 sentences (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 such 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.

2.3.3 OROMO STEMMERS

One of the stemmers for Oromigna language is the one developed by Wakshum (Wakshum, 2000). 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. Other linguistically valid suffixes are also included manually. The stemmer fined the longest suffixes that matches the end of a given word and remove.

Debela Tesfaye and Ermias Abebe (2010) have developed Oromigna stemmer by considering some problems from the stemmer developed by Wakshum M. and come out with new stemmer. This stemmer adopted some concepts from Porter stemmers. Specifically, Concepts about measure, arranging the rules in clusters, analyzing word formation based on the nature of their endings are taken from porter algorithm. This Afan Oromo stemmer is based on a series of steps that each removes a certain type of 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.

2.3.4 WOLAYTA STEMMER

Lemma L. (Lemma, 2003) 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.

2.4 STEMMING ALGORITHM FOR OTHER LANGUAGES

Stemmers are generally customized for each specific language. Their design requires some linguistic expertise in the language and an understanding of the needs of the specific information retrieval and natural language applications. Stemmers have been developed for a wide range of languages including French Language Savoy (1993), Slovene Language stemmer by Popovic & Willett (1992), Arabic Language stemmer by S. Khoja and R. Garside.(1999) and other many languages. The effectiveness of stemming across languages is varied and influenced by many factors.

2.4. 1 ENGLISH LANGUAGE STEMMERS

There are many kinds of stemming algorithm available for English language. These algorithms range from the simplest ones like the table look-up to the complicated ones performing iterative longest matching. Some of the English Language stemmers are developed by Lovins (1968), Dawson (1974), Porter (1980), Lennon et al. (1981), Paice (1990) and Frakes (1992).

2.4.1.1 LOVINS STEMMING ALGORITHM

The Lovins Stemmer is a single pass, context-sensitive, longest-match Stemmer developed by Julie Beth Lovins of Massachusetts Institute of Technology in 1968(lovins,1968). Lovins' paper was the first ever published description of a stemmer. It defines 294 endings, each linked to one of 29 conditions, plus 35 transformation rules. For a word being stemmed, an ending with a satisfying condition is found and removed. A suitable transformation rule is applied next. The aim of this step is to deal with doubled consonants and irregular plurals.

This early stemmer was aimed at both the IR and Computational Linguistics areas of

stemming. This stemmer, though innovative for its time, has the problematic task of trying to address two areas (IR and Linguistics) and cannot do well at either. The approach does not give good results with linguistics, as it is not complex enough to stem many suffixes due to their not being present in the rule list. There are problems regarding the reformation of words. This process uses the recoding rules to reform the stems into words to ensure they match stems of other similar meaning words. The main problem with this process is that it has been found to be highly unreliable and frequently fails to form words from the stems, or match the stems of like meaning words. The Stemmer does not satisfy from the IR viewpoint either, as its large rule set, and its recoding stage, affect its speed of execution. The Lovins Stemmer removes a maximum of one suffix from a word, due to its nature as single pass algorithm. It uses a list of about 250 different suffixes, and removes the longest suffix attached to the word, ensuring that the stem after the suffix has been removed is always at least 3 characters long. Then the ending of the stem may be reformed (e.g., by un-doubling a final consonant if applicable), by referring to a list of recoding transformations.

2.4.1.2 DAWSON STEMMING ALGORITHM

Dawson stemming algorithm (Dawson, 1974) is based on the one developed by Lovins (1968) which makes use of iterative longest match approval. It is a complex linguistically targeted Stemmer that is strongly based upon the Lovins Stemmer. Initially, Dawson uses the list that contains 260 English suffixes with associated removal condition codes given by Lovins. But, after he has corrected the list, he brings the total up to about 1200 suffixes. To avoid the problems of storage and processing time taken, the suffixes and their condition code numbers backwards are read, stored and indexes by length and final letter. Dawson does not use recording technique in his algorithm to handle stems and instead used an extension of the partial matching procedure also defined within the Lovins Paper. The basic principle of Dawson's algorithm is if two stem matches up to a certain number of characters and the remaining characters of each stem belong to the same stem ending class, then two stems are of the same form.

2.4.1.3 PORTER STEMMING ALGORITHM

The Porter Stemmer is a conflation Stemmer developed by Martin Porter at the University of Cambridge in 1980 (porter, 1980). The Stemmer is based on the idea that the suffixes in the English language (approximately 1200) are mostly made up of a combination of smaller and simpler suffixes. This Stemmer is a linear step Stemmer. Specifically it has five steps applying rules within each step. Within each step, if a suffix rule matched to a word, then the conditions attached to that rule are tested on what would be the resulting stem, if that suffix was removed, in the way defined by the rule. For example such a condition may be, the number of vowel characters, which are followed be a consonant character in the stem (Measure), must be greater than one for the rule to be applied.

Once a Rule passes its conditions and is accepted the rule fires and the suffix is removed and control moves to the next step. If the rule is not accepted then the next rule in the step is tested, until either a rule from that step fires and control passes to the next step or there are no more rules in that step whence control moves to the next step. This process continues for all five steps, the resultant stem being returned by the Stemmer after control has been passed from step five.

The objectives for the development of this stemming algorithm are to improve the performance of information retrieval and the success rate for the suffix discarding will be significantly less than 100% when the process is evaluated. The Porter algorithm consists of a set of condition rules. The conditions are divided into 3 classes; there are conditions on the stem, condition of the suffix and conditions on the rules. Porter's algorithm uses a dictionary of about 60 suffixes and has only a few context-sensitive and recording rules, and therefore is economical in storage and computing time.

2.4.1.4 PAICE/HUSK STEMMING ALGOTRITHM

The Paice/Husk Stemmer is a simple iterative Stemmer that removes the endings from a word in an indefinite number of steps (Paice C.1990). The Stemmer uses a separate rule

file, which is first read into an array or list. This file is divided into a series of sections, each section corresponding to a letter of the alphabet. The section for a given letter, say "e", contains the rules for all endings ending with "e", the sections being ordered alphabetically. An index can thus be built, leading from the last letter of the word to be stemmed to the first rule for that letter. When a word is to be processed, the stemmer takes its last letter and uses the index to find the first rule for that letter. If a rule is accepted then it is applied to the word. If it is not accepted, the rule index is incremented by one and the next rule is tried. However, if the first letter of the next rule does not match with the last letter of the word, this implies that no ending can be removed, and so the process terminates. Once a rule has been found to match, it is not applied at once, but must first be checked to confirm that it would leave an acceptable stem. When a rule is applied to a word, this usually means that the ending of the word is removed or replaced.

2.4.1.5 KROVETZ STEMMING ALGORITHM

The Krovetz Stemmer was developed by Bob Krovetz, at the University of Massachusetts, in 1993(Krovetz, 1993). It is quite a 'light' stemmer, as it makes use of inflectional linguistic morphology. The Krovetz Stemmer effectively and accurately removes inflectional suffixes in three steps, the conversion of a plural to its single form (e.g. ‘-ies’, ‘-es’, ‘-s’), the conversion of past to present tense (e.g. ‘-ed’), and the removal of ‘-ing’. The conversion process firstly removes the suffix, and then though a process of checking in a dictionary for any recoding, returns the stem to a word.

2.4.2 ARABIC STEMMING ALGORITHMS

Arabic is a highly inflected language and has a complex morphological structure. Some of the applications of Arabic natural language processing require the basic form of the word (root or stem) to be most effective, therefore stemming process is a necessity. There are several stemming approaches that are applied to Arabic language. The morphology complexity of Arabic makes it particularly difficult to develop natural language processing applications for Arabic information retrieval. In Semitic languages like Arabic, most noun, adjective, and verb stems are derived from a few thousand roots by

infixing and creating many variants of words (larkey, 2002). Arabic is highly productive, both derivationally and inflectionally. Definite articles, conjunctions, particles and other prefixes can attach to the beginning of a word, and large numbers of suffixes can attach to the end. A given headword can be found in huge number of different forms. Distributional analyses of Arabic newspaper text show empirically that there is more lexical variability in Arabic than in the European languages for which most IR and NLP work as been performed. Arabic text has more words occurring only once and more distinct words than English text samples of comparable size. The token to type ratio (mean number of occurrences over all distinct words in the sample) is smaller for Arabic texts than for comparably sized English texts (Khojo, 1999). Arabic orthography also contributes variability that can confuse information retrieval systems. It is not the right to left order of the characters, or the context-dependence of the appearance of the characters that are problematic. These are just rendering issues.

The factors described above make Arabic very difficult to stem. Several stemming algorithms for Arabic have been proposed based on different principles and each produces different sets of stem classifications. Larkey et al. gives a good summary of stemming approaches for the Arabic language [9].The most common approaches used in Arabic stemming are the light and the root-based stemmers. Root-based Stemming is based on removing all attached prefixes and suffixes in an attempt to extract the root of a given Arabic surface word. Several morphological analyzers have been developed for arabic, e.g. Khoja and Garside (Khoja 1999).

Light Stemming is used not to produce the linguistic root of a given Arabic surface form, but to remove the most frequent suffixes and prefixes. The most common suffixation includes duals and plurals for masculine and feminine, possessive forms, definite articles, and pronouns. Several light stemmers have been developed, all based on suffix and prefix removal and normalization. Examples of light stemmers include: Aljlayl & Frieder's Stemmer (S.Aljlayl) Darwish's Al-Stem(K.Darwish et al 2002) , and Larkey et al.'s U Mass Stemmer(L. S. Larkey et al.2001) .

All light stemmers adhere to the same steps of normalization and stemming. The main difference among them is the number of prefixes and suffixes removed from each one. During the normalization process, all diacritics, punctuation, and glyphs are removed. The light stemmers had different stopword lists consisting of Arabic pronouns, particles and the like removed after minimal normalization. Based on the test results of the stemmers proved that the light stemmer achieved superior performance over the root-based approach since it reduces sense ambiguity by grouping semantically related words into the same class. Although light stemming can correctly classify many variants of words into large stem classes, it can fail to classify other forms that should go together. For example, broken plurals for nouns and adjectives do not get conflated with their singular forms, and past tense verbs do not get conflated with their present tense forms, because they retain some affixes and internal differences.

2.5 EVALUATION METHODS FOR STEMMING ALGORITHMS

There are different methodologies employed to evaluate the performance of stemmers. The manual method, vocabulary reduction and Paice's method are identified as stemmer evaluation methods. In the manual method, a human being, who decides the correct stem for each word, performs the evaluation process. Three evaluation measurements are obtained in this manner: the number of correct results; the number of errors due to over stemming; and the number of errors due to under stemming. One of the purposes of stemmers is to reduce the size of the vocabulary for indexing purposes. The vocabulary reduction is obtained by dividing the number of words in the corpus by the number of stems generated, excluding repetitions.

Stemmers are evaluated using Paice's method based on error counting (paice, 1994). According to his method, measures of understemming and overstemming determine how good a stemmer is outside the retrieval context. In Paice's method, three measurements are implemented in order to make a qualitative comparison between different stemmers: the over stemming index (OI); the under stemming index (UI); and the stemming weight (SW). This method requires a word sampling, with no repetitions, separated into conceptual groups in which the words are semantically and morphologically related. The

SW is given by the ratio OI/UI. Paice has compared different English stemming algorithms in isolation from the context of an IR system and he did not use the traditional precision/recall parameters, an ideal stemmer should stem all words in a group to the same stem. If a stemmed group contains more than one unique stem, the stemmer has made understemming errors. In an IR system this corresponds to a negative effect on recall. If a stem of a certain group also occurs in other stemmed groups, the stemmer has made overstemming errors, which reduce precision. A good stemmer should therefore produce as few under and overstemming errors as possible.

CHAPTER THREE

MORPHOLOGY OF TIGRIGNA LANGUAGE

3.1 INTRODUCTION

Morphology is the branch of linguistics that deals with the internal structure of words and word formation, including affixation behavior, roots, and pattern properties (Spencer, 1991). Morphology is the main source of variation in natural language text, with suffixing and prefixing being the most common ways of creating a word variant. Morphology can be classified as either inflectional or derivational. Inflection is variation or change of form that words undergo to mark distinctions of case, gender, number, tense, person, mood, voice, comparison. Inflectional morphology is applied to a given stem with predictable formation. It does not affect the word's grammatical category, such as noun, verb, etc. Case, gender, number, tense, person, mood, and voice are some examples of characteristics that might be affected by inflection. Derivational morphology, on the other hand, concatenates to a given word a set of morphemes that may affect the grammatical and syntactic category of the word.

A word can have several word forms, e.g., the word "write" can take the forms "writes", "wrote" and "written", usually called inflected forms. The root is the original form of the word before any transformation process, and it plays an important role in language studies. The root is the form of a word from which the other forms can be derived using the morphological rules of a language. A morpheme is the smallest unit of a language that has a meaning and cannot be broken down further into meaningful or recognizable parts and should impart a function or a meaning to the word which they are part of. An affix is a morpheme that can be added before (prefix) or after (suffix), or inserted inside (infix) a root or a stem to form new words or meanings (Gregory, 2001). Morphological information of a language is useful for several natural language applications such as stemming, morphological analysis, text generation, machine translation, document retrieval, etc.

3.2 OVERVIEW OF TIGRIGNA LANGUAGE

Tigrigna is a member of the Ethio-Semitic languages, which belong to Afro-Asiatic super family (Voigt, 1987). Tigrigna is spoken primarily in Eritrea and Ethiopia. There are more than 6 million Tigrinya speakers worldwide. According to the 2007 population and housing census of thiopia, there are over 4.3 million Tigrigna speakers in Tigray (CSA, 2007) and according to Ethnologue there are 2.4 million Tigrigna speakers in Eritrea **(**Lewis, 2009**).** Tigrigna is written in the Ge'ez script which these days is called Ethiopic and originally developed for Geez language. In Tigrigna each symbol represents a consonant and vowel combination and the symbols are organized in groups of similar symbols on the basis of both the consonant and the vowel. For each consonant in each symbol, there is an unmarked symbol representing that consonant followed by a canonical or inherent vowel (Daniel, 2008).

Tigrigna Like other Semitic languages such as Arabic and Amharic exhibits a root-pattern morphological phenomenon. In addition, it uses different affixes to create inflectional and derivational word forms.

3.3 TIGRIGNA MORPHOLOGICAL SYSTEM AND WORD FORMATION

85% of the words in Tigrigna are created from a root of three radicals (trilateral words) and to a lesser extent there are also quadliteral, pen-literal, or hexa-literal words (Kasa G., 2004). Each word group generates an increased verb forms and noun forms by the addition of derivational and inflectional affixes. Words in Tigrigna are built from the roots by means of a variety of morphological operations such as compounding, affixation, and reduplication (Amanuel, 1998).

An affix in Tigrigna is a morpheme that can be added before or after, or inserted inside, a root or a stem as a prefix, suffix or infix, respectively, to form new words or meanings. Tigrigna affixes have the feature of concatenating with each other in predefined linguistic rules. This feature increases the overall number of affixes (Kasa G., 2004). There are also some prefixes and suffixes which determine whether a word is a subject marker, pronoun, preposition, or a definite article. Tigrigna is highly productive, both derivationally and

inflectionally. Definite articles, conjunctions, particles and other prefixes can attach to the beginning of a word, and large numbers of suffixes can attach to the end. A given headword can be found in huge number of different forms.

Tigrigna concatenative morphology regulates how a stem and affixes glue together, while non-concatenative one combines morphemes in more complex ways. Affixes in Tigrigna can be classified as four categories (Kasa G., 2004). Prefixes precede the base form, such as *`Intey-,`ay-, kemz-, ktete-, sle-, zte-* . Suffixes follow the base form, i.e. *-kum, -tat, -tatat, -net, -awi* and Infixes are inside the base form. Circumfixes are affixes attached before and after the base form at the same time. While circumfixes formally are combination of allowed prefixes and suffixes, they have to be treated as discontinuous units for semantic and grammatical reasons. Tigrigna non-concatenative morphology refers to reduplicated morpheme forms. Reduplicated words based on morpheme regularity are grouped into full reduplication (e.g., the word ስትይስትይ is derived from the stem ስትይ) and partial reduplication of different kinds. The latter includes reduplicated stems with affixes (e.g. word ሰባበረ is derived from stem ሰበረ sebere, ተረ‚ጋገሙ ‘teregagemu’ is derived from stem ረገመ ‘regeme’, the word ገልጠምጠም ‘gelTemTem’ is derived from the stem ገልጠም ‘gelTem’) and there also various irregular reduplications.

3.4 DERIVATIONAL AND INFLECTIONAL MORPHOLOGY

There are five parts of speech in Tigrigna: adjectives, nouns, verbs, adverbs, and prepositions (Daniel, 2008). Prepositions and conjunctions are totally unproductive. Adverbs are few in number and are less productive. Therefore, the discussion of derivational and inflectional morphology concentrates on the remaining three parts of speech, namely verbs, nouns, and adjectives.

3.5 INFLECTIONAL MORPHOLOGY OF TIGRIGNA

As Tigrigna is a highly inflectional language definite articles, conjunctions, particles and other prefixes can attach to the beginning of a word, and large numbers of suffixes can attach to the end. A given root of word can be found in huge number of different forms.

3.5.1 INFLECTION OF VERBS

This section presents the inflection of verbs. It is compiled for the purpose of the study from Tigrigna grammar books by Daniel Teklu (2008) and Kasa G. (2004).

A significantly large part of the vocabulary consists of verbs, which exhibit different morphsyntactic properties based on the arrangement of the consonant-vowel patterns. For example, the root sbr, meaning ’to break’ can have the perfect form sebere with the pattern CVCVCV, imperfect form *tsebr* with the pattern CCVCC, gerund form sebirka with the pattern CVCVCCV, imperative form sber with the pattern CCVC, causative form ’asbere with the pattern as-CVCV, passive form tesebere with the pattern te-CVCVCV,etc. Subject, gender, number, etc are also indicated as bound morphemes on the verb, as well as objects and possession markers, mood and tense, transitive, dative, negative, etc, producing complex verb morphology.

The simplest form of the verb is the third person masculine singular of the perfect tense. In most Tigrigna dictionaries, all the words derived from a triliteral root are entered under the third person masculine singular form of the verb. Each three-consonant (or "triliteral") root belongs to one of three conjugation classes, conventionally known as A, B, and C. This division is a basic feature of Ethiopian Semitic languages.

Most three-consonant roots are in the A class. In the citation form (perfect), these have no germination but the vowel ‘*e*’ appears between both pairs of consonants. Examples are: ደረፈ derefe “he sung”, ደየበ deyebe “he climbed”, ሰተየ seteye “he drank”. The B class is distinguished by the gemination of the second consonant in all forms. Some Examples are: ደቀሰ deqqese ‘sleep’ ወሰኽ wesseKe ‘add’. The relatively few members of the C class take the vowel *a* between the first and second consonants. Examples are ባረኽ bareKe 'bless' and ናፈቐ nafeqe 'long for, miss'.

Tigrigna also has a significant number of four-consonant (or "quadriliteral") roots. These fall into a single conjugation class. Examples are መስከረ meskere 'testify' and ቀልጠፈ qelTefe 'hurry’. The language also has five-consonant (or "quinquiliteral") roots. Most, if

not all, of these are "defective" in the sense that their simplest form takes the te- prefix. Examples are ተንቀጥቀጠ te-nqetkeTe 'tremble' and ተምበርከኸ *te-mberkeKe* 'kneel'.

Tigrigna verbs have two tenses: perfect and imperfect. Perfect tense denotes actions completed, while imperfect denotes uncompleted actions. The imperfect tense has four moods: indicative, subjective, jussive, and imperative. Tigrigna verbs in perfect tense consist of a stem and a subject marker. The subject marker indicates the person, gender, and number of the subject. The form of a verb in perfect tense can have subject marker and pronoun suffix. The form of a subject-marker is determined together by the person, gender, and number of the subject. Other elements like negative markers also inflect verbs in Tigrigna.

As in other Semitic languages, Tigrigna verbs are very complex consisting of a stem and up to four prefixes and four suffixes. The stem in turn is composed of a root, representing the purely lexical component of the verb, and a template, consisting of slots for the root segments and for the vowels (and sometimes consonants) that are inserted around and between these segments. The template represents tense, aspect, and mood.

Each lexeme can appear in four different tense-aspect-mood (TAM) categories, conventionally referred to as perfective, imperfective, jussive/imperative, and gerund. For example, the verb ayftewn ‘he is not liked’ has the lemma tefetewe ’he was liked’, which is derived from the verb root ftw. Every Tigrigna verb must agree with its subject. As in other Semitic languages, subject agreement is expressed by suffixes alone in some TAM categories (perfective and gerundive) and by a combination of prefixes and suffixes. Tigrigna verbs may also have a suffix representing the person, number, and gender of a direct object or an indirect object that is definite.

Tigrigna verbs are inflected for person, gender, number, and time with basic verb form being the third person masculine singular. Tigrigna verbs have two tenses: perfect and imperfect. Perfect tense denotes actions completed, while imperfect denotes uncompleted actions. The imperfect tense has four moods: indicative, subjective, jussive, and imperative. Tigrigna verbs are conjugated in perfective, imperfective, indicative,

subjective, jussive, and imperative. In conjugating the verbs affixes are attached to the verbs.

3.5.1.1 INFLECTION OF PERFECTIVE TENSE

The perfect tense which is the basic form normally expresses the past tense and consist of a stem and a subject marker. The form of a verb in perfect tense can have subject marker and pronoun suffix. The subject marker indicates the person, gender, and number of the subject. The form of a subject-marker is determined together by the person, gender, and number of the subject (Daniel T., 2008). The following example demonstrates how suffixes are attached to verbs for indicating subject marker and pronoun.

Verb Variations	**Person**			**Gender**		**Number**	
	1	**2**	**3**	**M**	**F**	**Singular**	**Plural**
ቀቲለ	+			+	+	+	
ቀቲልና	+			+	+		+
ቀቲልካ		+		+		+	
ቀቲልኪ		+			+	+	
ቀቲልኩም		+			+		+
ቀቲልክን		+					
ቀቲላ			+		+	+	
ቀቲለን			+		+		+
ቀቲሎም			+	+			+
ቀተለት			+		+	+	
ቀቲሉ			+	+		+	

Table 3. 1: Inflections perfect tense

The suffixes attached are አ/a/, ና/na/, ካ/ka/, ኪ/ki/, ኩም/kum/, ክን/kn/, ኡ/u/, ኣ/a/, አን/en/, አም/om/, አት/et/. Suffixes of Subject-Marker and pronoun indicators are used commonly without any variation with verbs of type A, B, C and quadriradicals.

3.5.1.2 INFLECTION OF IMPERFECTIVE TENSE

The imperfect tense has four moods: indicative, subjective, jussive, and imperative and is inflected by prefixing and suffixing gender, person, and number morphemes to the imperfective verb stem. The table below shows how suffixes and prefixes are added for the root verb "ገበረ"

Person	Singular	Plural
1st person	እገብር (`I-gebr)	ንገብር (n-gebr)
2nd person-masculine	ትገብር (t-gebr)	ትገብሩ (t-gebr-u)
2nd person- feminine	ትገብሪ (t-gebr-i)	ትገብራ (t-gebr-a)
3rd person-masculine	ይገብር(y-gebr)	ይገብሩ (y-gebr-u)
3rd person-feminine	ትገብር (t-gebr)	ይገብራ (y-gebr-a)

Table 3. 2: Inflection of Imperfective tense

In imperative tense prefixes እ(`I-), ት(t), ይ(y),ን(n) and the suffixes attached are ኡ(u), ኢ(i), ኣ(a). To indicate negative verbs the morphemes ኣይ/ay/, ኣይት/ayt/, ኣይን/ayn/ are added as prefixes and ን/n/, ኣን/an/, ኡን/un/ are added as suffixes.

3.5.1.3 INFLECTION OF GERUNDIVE FORM OF VERB

Person	Singular	Plural
1st person	ነቢረ(nebir-e)	ነበርና (neber-na)
2nd person-masculine	ነበርካ (neber-ka)	ነበርኩም(neber-kum)
2nd person- feminine	ነበርኪ (neber-ki)	ነበርክን (neber-kn)
3rd person-masculine	ነቢሩ(neber-u)	ነበሩ (neber-u)
3rd person-feminine	ነቢራ (nebir-a)	ነቢረን (nebir-en)

Table 3. 3: Inflection in Gerundive form of verb

To make the gerundive form of the verb ኧ/e/, ካ/ka/, ኪ/ki/, ኡ/u/, ኣ/a/, ና/na/, ኩም/kum/, ክን/kn/, ኧን/en/ morphemes are attached to the root word.

3.5.1.4 INFLECTION OF IMPERATIVE FORM OF VERB

Person	Singular	Plural
1st person	ክፅሕፍ (kSHf)	ንፅሕፍ (nSHf)
2nd person-masculine	ትፅሕፍ (tSHf)	ትፅሓፉ(tSHafu)
2nd person- feminine	ትፅሕፊ (tSHfi)	ትፅሕፋ(tSHfa)
3rd person-masculine	ይፅሓፍ (ySHaf)	ይፅሓፉ(ySHfu)
3rd person-feminine	ትፅሓፍ (tSHaf)	ይፅሓፋ(ySHafa)

Table 3.4: Inflection of Imperative form of verb

3.5.2 INFLECTION OF NOUNS

Tigrigna nouns inflect for case, number, definiteness, and gender (Daniel Teklu, 20068). A noun has the nominative case when it is a subject; accusative when it is the object of a verb; and genitive when it is the object of a preposition. The form of Tigrigna noun is determined by its gender, number, and grammatical case. Most plural nouns are formed by adding a plural marker affix (-*tat* or -at) to the singular form. Although when referring to groups belonging to a certain tribe or country –*yan* is affixed. There are a set of affixes that are used to make plural nouns and are attached as prefix or suffixes to the nouns. The affixes -ታት/-tat/, ኣት/-at/, ኣን/-an/, ኦት/-ot/, ውቲ/-wti/, ቲ/-ti/ are used as suffixes to inflect nouns. Here are some examples to show the inflection of nouns.

Noun	Noun-suffix	After suffixation
ባሕሪ	ባሕሪ-ታት	ባሕሪታት
እምባ	እምባ-ታት	እምባታት
ስእሊ	ስእሊ-ታት	ስእልታት
ሃገር	ሃገር-ኣት	ሃገራት
ሰማይ	ሰማይ-ኣት	ሰማያት
ምእመን	ምእመን-ኣን	ምእመናን
መምህር	መምህር-ኣን	መምህራን

Table 3. 5: Inflections of Nouns

Another form of inflection for nouns is created by attaching the morpheme ኣ/a-/ as prefix. For example in the words ገረብ(gereb, forest), ፈረስ(feres, horse), ከረን(keren, mountain), ኣግራብ(agrab, forests), ኣፍራስ(afras, horses), ኣከራን(Akran, mountains).

Tigrigna has two grammatical genders: masculine and feminine, and all nouns belong to either one or the other and inanimate objects are take one of the genders. Some noun pairs for people distinguish masculine and feminine by their endings, with the feminine signaled by ኢት/it/ and the masculine signaled by ኢ/i/. These include agent nouns derived from verbs — ከፈተ kefete 'open', ከፋቲ kefati 'opener (m.)', ከፋቲት kefatit 'opener (f.)' — and nouns for nationalities or natives of particular regions - ትግራዋይ tgraway 'Tigrean (m.)', ትግራወይቲ tgraweyti 'Tigrean (f.)'.

3.5.3 INFLECTION OF ADJECTIVES

Tigrigna adjectives inflect for number and gender. Tigrigna adjectives may have separate masculine singular, feminine singular and plural forms, and adjectives usually agree in gender and number with the nouns they modify (Daniel T., 2008). The plural forms follow the same patterns as noun plurals; that is, they may be formed by suffixes or internal changes or a combination of the two. The affixes that are used for the inflections of the adjectives are ኦ/o/, ቲ/ti/, ኣት/at/, ኣን/an/ and ኦት/ot/. Table 3.6 shows inflection adjectives for number.

Singular Adjectives	Plural Adjectives	Affix attached
SheKali	SheKalo	-o
Kedani	Kedano	-o
Mehazi	Mehazti	-ti
Qetal	Qetelti	-ti
Kbur	Kburat	-at

Senef	Senefat	-at
Harestay	Harestot	-ot
Zebenay	Zebenot	-ot

Table 3. 6: Inflection of Adjectives

Adjectives are also inflected for gender by adding the infix –a- and suffix –ti. For example, words such as qeyaH(fm) , qeyaHti and qeyh(m), qetan(fm) ,qetenti and qetin(m) ,belaH(fm), belaHti and beliH (fm) are inflected forms of the qyH,qtn,blH respectively.

3.6 DERIVATIONAL MORPHOLOGY

Derivational morphology describes how affixes combine with word stems to derive new words. Derivational affixes may affect the part-of-speech and meaning of a word.

3.6.1 DERIVATION OF VERBS

Unlike the other word categories such as nouns and adjectives, the derivation of verbs from other parts of speech is not common. Almost all Tigrigna verbs are derived from root consonants, as indicated by Daniel (2008). Traditionally a distinction is made between simple and derived verbs.

Simple verbs are those verbs derived from roots by intercalating vowel patterns whereas derived verbs are considered as derivatives of simple verbs. The derivation process can be an internal one in which consonant-vowel patterns are changed, an external one where derivational affixes are attached to the simple derived verbs or a combination of the internal and external derivational processes. The derivation of causative, passive, repetitive and reciprocal verbs is presented below.

1. Causative: Causative verbs are derived by adding the derivational morphemes ‘a- and to the verb stem as in the examples በፅሐ /beSHe-/ ‘arrive’ - ኣበፅሐ- /‘abSHe/ ’cause to arrive’ and ወሰደ /wesed/ ’take’ ኣውሰደ /awsede-/ ’cause to take’. In most cases the ‘*a*-morpheme is used to form causative of intransitive verbs, transitive ones and verbs of

state. Some exceptions are the verbs that begin with ‘a, always take the morpheme ‘a but add the morpheme I after the morpheme ‘a to form causative e.g. ኣሰረ /‘asere/, ኣእሰረ /‘aIsere /.

2. Passive/Reflexive: The passive verbs are derived using the derivational morpheme ተ /te/. This derivational morpheme is realized as ተ-/te-/ before consonants and as ት-/t-/ before vowels. Moreover, in the imperfect, jussive and in derived nominals like verbal noun, the derivational morpheme ት/t-/ is used. In this case, it assimilates to the first consonant of the verb stem, and as a result, the first radical of the verb geminates. Some exceptions are intransitive verbs like ፈሊሑ /faliHu/ ’it boiled’ that form their passive forms using the prefix ተ- /te-/ as in ተፈሊሑ /tafeliHu/ ’it was boiled’. Such kind of verbs can derive their passive from their causative form (ኣፍሊሑ/afliHu/’he boiled’).

3. Reduplicative/repetitive: Reduplicative stems indicate an action which is performed repeatedly. For tri-radical verbs, such stems are formed by duplicating the second consonant of the root and using the ኣ-/a-/ after the duplicated consonant as in ሰባበረ /se-ba-bere/ ’he broke repeatedely’ derived from the root ሰብር/sbr/ break. All verb types, Type A, B and C have the same reduplicative forms.

4. Reciprocal: Reciprocal verbs are derived by prefixing the derivational morpheme ተ- /te-/ either to the derived type C forms (that use the vowel a after the first radical) or to the reduplicative stem. For example, reciprocal forms of ተቃተሉ /teqatelu/ ’killed each other’ and ተቀታተሉ /teqetatelu/ ’killed one another’ are derived from the derived type C stem qatelu- and reduplicative stem qetatelu-, respectively. The causative of reciprocal verbs are formed by adding the causative prefix ’a- to the reciprocal verb forms. However, the reciprocal verb prefix t- or ‘a- assimilates to the stem-initial consonant (thus causes the first radical of the stem to geminate) and does not show up in the surface form of the reciprocal causative.

3.6.2 DERIVATION OF NOUNS

Tigrigna nouns can be either primary or derived. They are derived if they are related in their root consonants and/or meaning to verbs, adjectives, or other nouns. Otherwise, they are primary. For example, a noun እግሪ /Igri/ ’foot, leg’ is primary but, እግረኛ/IgreNa/ ’pedestrian’ is derived from the nominal base Igri by adding the morpheme –‘eNa.

Nouns are derived from other nouns, adjectives, roots, stems, and the infinitive form of a verb by affixation and intercalation. The morphemes -ነት/-net/, -ኢት/-`it/, -ኣት/-`at/,-ኡት/ut/, -ትኦ/-to/, -ኦ/o/, -ኢ/i/, -ኣ/a/, -ኣን/an/, -አኛ/-‘eNa/, -ኛ/-Na/, -አት/-et/, ኣዊ/-awi/, -ተኛ/-teNa/, -ና/-na/ and the prefix መ-/me-/ are used to derive nouns from other nouns. From the adjectives, nouns can be derived using the suffixes /net/ and /-et/ as in the examples /merzamanet/ ’generosity’ which is derived from the adjective /merzam/ ’poison’ and flTet ’knowledge’ from the adjective fluT ’known’. Nouns can also be derived from verbal roots by intercalation and affixation. **Table 3.6** shows some examples of derived nouns from other nouns.

Base form	**Bound morpheme**	**Derived noun**
Mskr	-net	mskrnet
Selam	-awi	selamawi
Areb	-Na	ArebNa
Xlm	-at	Xlmat
bSh	-it	bSHit
Whb	-to	whbto
Dfn	-o	dfno
Hlm	-i	Hlmi
Lmn	-a	lmena
Qtl	-et	qtlet
sKr	-an	sKran
gzI	me-,-ti	megzaIti
srH	-teNa	seraHteNa
Merkeb	-eNa	merkebeNa

Gbr	-na	gbrna

Table 3.7: Nouns derived from other nouns

In Tigrigna, nouns can also be formed through compounding. For example, ቤትብልዒ ‘restuarant’ is derived from the nouns ቤት /bet/ ‘house’ and ብልዒ /bl`i/ ’food’. As it can be seen, no morpheme is used to bind the two nouns. But, there are also compound nouns whose components came together by inserting the compounding morpheme ኣ/ `e/ as in ቤተክርስትያን /betekrstyan/ ’church’ which is formed from ቤት/bet/ ’house’ and ክርስትያን/krstyan/ ’Christian’.

3.6.3 DERIVATION OF ADJECTIVE

Adjectives in Tigrigna include all the words that modify nouns and can be modified by the word ብጣዕሚ/btaimi/ ’very, greatly’ (Daniel T., 2008). As it is true for nouns, adjectives can also be primary (such as ለዋህ /lewah/ ’kind’) or derived, although the number of primary adjectives is very small. Adjectives are derived from nouns, stems or verbal roots by adding a suffix and by intercalation. The suffixes ኣም/-am/, -ዊ /-wi/, -ኣዊ /-awi/, -ኣይ/-ay/, -ኣታይ/-atay/,-ታይ/-tay/, -ኣኛ/-eNa/ and -ኣዋይ/-away/ are used in the derivation of adjectives from nouns. For example it is possible to derive ሃፍታም/haftam/’rich, wealthy’, ተንኮለኛ/tenkoleNA/, ዘበናዊ/zebenawi/’modern’andማእኸላይ/maIKelay/ ‘central’ from the nouns ሃፍቲ/hafti/ ’wealth’, ተንኮል/tenkol/ ’’, ዘ በን/zeben/ ’period’ and ማእኸል /maIKel/ ’center’, respectively. Adjectives can also be derived either from roots by intercalation of vocalic elements or attaching a suffix to bound stems.

CHAPTER FOUR

DESIGN AND IMPLEMTATION OF THE STEMMER

4.1 THE CORPUS

The researcher has utilized different sources of text for the development of the stemmer and the experiments. Since there is no document collection for Tigrigna language available like the TREC, and the CLEF collections for the English language, the researcher used his own collections. The researcher took documents from different sources and organizes them into three collections. The first corpus consists a collection of newswire articles from three popular online Tigrigna newspapers called Mekalih Tigray, Woyn and Hadas Eritrea. The articles covered topics such as Politics, economics, general, religion, science, medical, sport, and art. The topics are varied so as to represent the different morphological variation of words. The first corpus consisted of 76,515 word tokens and 17,634 distinct word types. The second corpus consists sample words taken from two Tigrigna books called Gahdi-1 and Gahdi-2. The second corpus consisted of the 9245 word tokens and 3375 distinct word types. The third corpus is collected from different sources (such as bible, fictions and websites) selected by the researcher to represent the morphological complexity of the language. The third corpus consisted of 44,730 word tokens and 13,135 distinct word types. Table 4.1 shows the statistics of each collection.

Name	**Description**	**Word tokens**	**Word types**
Newspapers	A document collection of three news papers(Woyn,MekalihTigray, Hadas Eritrea)	76515	17634
TIGBooks	Two Tigrigna books (Gahdi 1 and 2)	9245	3375
Generaldoc	A document collected from different sources	44730	13135

Table 4. 1: Corpus used for the development of stop word list and affixes

This corpus was used to collect word frequency and affixes attached to a known stem.

4.2 MORPHOLOGICAL PREPROCESSING

The stream of characters in a natural language text must be broken up into distinct meaningful units before any language processing can be performed. Preprocessing is an important part of all text processing. In the preprocessing stage file formats, character sets, and variant forms can be converted, so that all text, regardless of its source, is in the same format. In later stages all further processing can then be consistently applied to all of the data, without the need to handle exceptions. Preprocessing must ensure that the source text be presented to NLP in a form usable for it. For example, NLP programs usually need their input to be tokenized, i.e. text elements usually word forms or sentences are identified and placed on separate lines of the input (Kaplan, 2005). In the preprocessing stage this study addresses tokenization, normalization and transliteration and stopword removal.

4.2.1 TOKENIZATION

In this study, words are taken as tokens. All punctuation marks, control characters, numbers and special characters are removed from the text before the data is processed. All punctuation marks are converted to space and space is used as a word demarcation. Hence, if a sequence of characters is followed by space, that sequence is identified as a word. A consecutive sequence of valid characters was recognized as a word in the tokenization process.

4.2.2 NORMALIZATION

Different symbols in Tigrigna writing system with the same sound are available. These different symbols must be considered as equivalent because they do not cause changes in meaning. As a result, in this research, all different symbols of the same sound were converted to one common form. For example, the characters ሠ and ሰ have similar sound (with the sound se). These two characters with equivalent sound are converted to ሰ (se). ፀ and ጸ are characters with equivalent sound and are changed to ፀ (tse).

4.2.3 TRANSLITERATION

The document collection used in this research is published using the Ethiopic script and using a variety of fonts. In addition to this Tigrigna characters fuse consonant and vowels in to one character. In order to simplify the analysis and to have a unified representation of the texts, transliteration of all Tigrigna texts into Latin characters is necessary. In this paper SERA is used to convert Tigrigna text to Latin equivalent. SERA is a system for ASCII representation of Ethiopic characters. While there is no single agreed-on standard for converting Ge'ez script to Latin text, the SERA transcription system (Firdyiwek and Yaqob, 1997), which represents Ge'ez characters using ASCII characters is common in computational work on Ge'ez script and is used in this paper.

4.3 CONSTRUCTION OF GENERAL PURPOSE STOPWORD LIST

Tigrigna domain independent stopwords include prepositions, conjunctions, and articles. General Stopwords are words which serve no purpose for NLP applications, but are used very frequently in composing documents, and these stopword lists are developed for two main reasons: These words may damage the stemming performance because stemming stopwords will not have any advantage for any NLP application. Secondly, removing stopwords helps to reduce the size of the file. Stopping is the act of removing words that do not contribute much to the content of the documents (Baeza-Yates and Ribeiro-Neto, 1999).As in other languages, Tigrigna also contains stop words. The articles of Tigrigna "Iti", "Ita", and "Itom" and the conjunctions "kab", "ab", and "nab" are examples of such highly frequent words. Removing Tigrigna stopwords can reduce file size and processing time.

For the purposes of this research, a Python program was written to generate a Tigrigna stopword list consisting of pronouns, prepositions, particles and articles. In establishing a general stopword list for Tigrigna, the researcher followed the guidelines described by Fox (1990). Firstly, all the word forms appearing in the collection of Tigrigna documents are sorted according to their frequency of occurrence and the 1000 most frequently occurring words are extracted. Secondly, this list was inspected manually to remove all

verbs, nouns and adjectives more or less directly related with the main subjects of the underlying collections. For example, the words "hzbi" ranked at the 13th position on the list as well as the noun "sraH" ranked at the 16th position and "tigray" ranked at the 40th position were removed from the list. Other nouns such as "mengsti" (government), "wereda", "merEt", "kunetat", "sr`at" and adjectives such as "senay", "fluy" were also removed. Thirdly, some non-information-bearing words were included manually by the researcher even if they did not appear in the first 1000 most frequent words. For example, various personal or possessive pronouns such as "ane" (me), "natka" (yours), prepositions "dHri" (after) and conjunctions "btewesaki" (in addition) were added.

The general stopword list compiled for Tigrigna contains 540 words and this list is included in Appendix 1. Ordering the words according to their occurrence frequency confirms Zipf's law (Zipf, 1949). Zipf's law states that when the distinct words in a text are arranged in decreasing order of their frequency of occuerence (most frequent words first), the occurence characterstics of the vocabulary can be characterized by the constant rank-frequency law of Zipf:

Frequency * Rank = constant

that is if the words, w, in a collection are ranked, r, by their frequency, f, they roughly fit the relation: r * f = c . For Tigrigna document collection zipf's law assumption gives the correct results.

Based on the corpus used to extract stop words, the 10 most frequent words represent 10.29% of all occurrences in this text collection, while the 30 most frequent words cover 14.8% of all forms appearing in the documents. When using such a stopword list, the size of the file was reduced by 23% for the collection. The text collection used for compilation of the stopword list contains a total number of 120897words with 26429 numbers of unique words. Table 4.2 shows the 25 most frequent Tigrigna words from the document collection.

Rank	Words	Frequency of Occurrence of word
1	Ab	3474
2	Iyu	1837
3	Nay	1692
4	Kab	1048
5	Iti	1103
6	Izi	812
7	Iwn	680
8	Kem	645
9	Dma	640
10	Nab	511
11	Zelo	432
12	Hade	430
13	Ms	423
14	Iyom	396
15	Neti	329
16	ke`a	283
17	Kulu	276
18	Alo	270
19	Gn	268
20	Koynu	256
21	Abti	246
22	Itom	238
23	Wn	235
24	Abzi	233
25	Nezi	230

Table 4.2:Most frequently occurring Tigrigna words

4.4 COMPILATION OF TIGRIGNA AFFIXES

The Tigrigna affixes consist of four different types, which are the prefix, suffix, prefix-suffix pair, and infix. Unlike English stemmers which work quiet well just by removing suffixes alone to obtain the stems, an effective and powerful Tigrigna stemmer not only must be able to remove the suffixes, but also the prefixes, prefix-suffix pairs, and infixes as well. Without removing all these affixes, the stemmer cannot be effectively used to stem Tigrigna documents. The question that arises here is that which is the best order of the affixes to be applied in terms of producing the minimum number of stemming errors.

4.4.1. COMPILATION OF PREFIXES

A set of prefixes that are used to develop the algorithm is compiled from different sources based on the grammatical functions of the affixes and their occurrence frequencies among the Tigrigna words found in the document collection. The list is collected from two Tigrigna grammar books by Kassa G. (2004**)** and Daniel (2008), a stemmer developed by Girma (2001) and from the document collection used for this study. The prefixes list ranges from single prefixes such as "b", "bz", "zey", "`Inte", "kem", "te" ","sle" to combinations of prefixes such as "`Intezey", "`Intezeyte", "bzey", "kemzey","kemzeyte","slezey".Table 4.3 shows some of the prefixes collected for the development of the algorithm. The complete list of prefixes is given in appendix 2.

Sample of single prefixes	**Combinations of prefixes**
Key	Slezte
Zte	`Inteyte
Zeyte	`Indate
`Inda	Kemzte
Intey	`IndHrzey
`IndHr	`Inkeyte
Tey	Slezeyte

Table 4. 3: Sample prefixes of Tigrigna

4.4.2 COMPILATION OF SUFFIXES

This is a list of suffixes that is used in the stemmer. A similar approach as that used to compile the list of prefixes is used to develop the list of suffixes. The suffix list ranges from single suffixes such as "ki", "ka", "kum", "om", " na", " lom"," tat" to combinations of prefixes such as "kumwen", "kana", "Natat", " tatomn"," tatkum","awinet". Table 4.4 shows some of the suffixes collected for the development of the algorithm. The complete list of suffixes is given in appendix 3.

Single suffixes	**Combinations of suffixes**
Net	Knaley
Ay	Kumley
Ku	Kayom
Ley	Waynet
Lu	Kumna
Yo	Tatnan
To	Kiyen
Net	Knalna
Kn	teNatat
tn	Tlna

Table 4. 4: Sample suffixes of Tigrigna

4.5 THE RULES

Trying to deal with each affix individually, the following rules are created. The rules are presented below in pseudo-code:

Rule-set 1

if (*word begins with `IndHrzeyte | `IndHrzeyt****)***

*remove the prefix****;***

Rule-set 2

if (*word begins with kemzeytete | slezeytete | `IndHrzeyt)*

*remove the prefix****;***

Rule-set 3

if (*word begins with `IndHrzey****)***

*remove the prefix****;***

Rule-set 4

if (*word begins with kemzeyte | `Inteyte* ***)***

*remove the prefix****;***

Rule-set 5

if (*word begins with `Intete | `Indate | slezeye |slezeyt)*

*remove the prefix****;***

Rule-set 6

if (*word begins with kemzey | kem`It |kemzte|mstete|slezte |slezey|`Inate |`Intez' |`IndHr****)***

*remove the prefix****;***

Rule-set 7

if (*word begins with kemte | kemIn | keyte | `Inte | zeyte | kemze | ztete | ktete | slete | sleze | `aymte| `Intey | `Inkey | ksabz | `ayte | `Inda| bzeym****)***

remove the prefix;

Rule-set 8

if (*word begins with kemz | slet |`Ite |zete |zeym | kemt | ztet | `Ina |keye | `Int | ztet | mste |kndi | ktet |slez |kndt | kndi| ksab|nzte|nkey |bate | bzey | bebi | bzte| n`kn* ***)***

*remove the prefix****;***

if *(word begins with `ayt) and the next letter is vowel*

remove [`ay];

else

remove the prefix;

if *(word begins with `ayn) and the next letter is vowel*

remove [`ay];

else

remove the prefix;

Rule-set 9

if (*word begins with tey | nze |bte |nme |b`a |bmt |ayn |`It |key|beb |`ay |*
*`In | zte |nen| nkn | kne| sle |kte|kem |zey****)***
*remove the prefix****;***

Rule-set 10

if (*word begins with ze |te|ti |`I |kt |ke |kn |nm| nz| nb |nk |bz |bz|bb| ms| mt |****)***
*remove the prefix****;***
if *(word begins with bm) and the next letter is vowel*
remove [b];
else
remove the prefix;

Rules for removing suffixes

Rule-set 1

if (*word ends on teNatat |kumleyn |knaleyn |aynetom****)***
*remove the suffix****;***

Rule-set 2

if (*word ends on knalom |kumwom | Kumwom |tatnan |tatomn |kalomn |nayomn |awinet |waynet | netawi | tatkum | nalkum |kulkum |kumley |kaleyn |knaley |knalna |kumlen |knalen |kumwen ****)***
*remove the suffix****;***

Rule-set 3

if (*word ends on tatom |kalom |nalom |kayom |Kayom|klomn |Natat |titat |winet |nakum |atkum |tlkum |kleyn |nayen|kaley|atnan |naley|wetay |tatna |knana |Knana |kumna |Kumna |knalu |ynetn |kumni |Kumni |knani |Knani |nalki | klom |kulki |nalen |kalen | nalka| kulka | knala | kiyen | kayen | nayom |tatat****)***
*remove the suffix****;***

Rule-set 4

if (*word ends on atom |tatn |etom| otat | klom |tlom |lomn |knom |ttat |ynet |tkum | ukum |uwon |uwom |kley |mley |tley |atay |away |kana |Kana |nana |tlna | klna |tatu |netu | nalu |kalu | netn |Kani |kani |nani |omni |umni |atni |atna |naki | tlki | klen |tlen |atkn |naka |tlka |Kila |nala |kula |kyen |aten |uwen | knen |Knen |elom| nayo ****)***

*remove the suffix**;***

Rule-set 5

if (*word ends on tat |uwo |atn |atu |tom |lom |wom |yom |omn |net |awi |tla | Kum |kum |kWa |ley |tay |una |kna |tna |Kna |yen | mlu |tlu |klu |knu |nan | uni |kni |len |ukn|ti* ***)***

*remove the suffix**;***

Rule-set 6

if (*word ends on om|Ka |en |wo |at |ot |on |et |Ki |na |ki |Kn |tn |Ku |ey |ay | na|ku|tn |kn |Ka* ***)***

*remove the suffix**;***

Rule-set 7

if (*word ends on u|i |a|o* ***)***

*remove the suffix**;***

4.6 THE PROPOSED ALGORITHM

Table 4.5 below shows the proposed stemmer.

1. *Get the word and IF word is in stop word list*
   Remove word
2. *Get the word and count the number of radicals*
3. *IF the number of radicals is < 3*
   Return word
4. *IF the number of radicals >=3*
   APPLY the specific rules
5. *Get the word and count the number of radicals*
6. *IF the number of radicals is < 3*
   Return word
7. *IF the number of radicals >=3*
   APPLY the specific rules

Table 4.5:The proposed algorithm

The algorithm for the stemmer is described in detail as follows:

Step 1. Prefix Stemming

In the prefix stemming rules, 10 groups of prefixes are identified ranging from one-letter prefixes to ten-letter prefixes. The system starts stemming the words in the word lists from the longest prefixes (ten-letter prefixes) to the one-letter prefixes. The program reads words from text file and if the word is found in the stop word list, it is excluded from prefix stemming. Otherwise it adheres the prefix removal procedures according to the specific rules provided.

Step 2. Suffix Stemming

In the suffix stemming rules, 7 groups of suffixes are identified ranging from one-letter suffixes to seven-letter suffixes. The system starts stemming the words in the word lists from the longest suffixes (seven-letter suffixes) to the one-letter suffixes.

4.7 IMPLEMENTATION OF THE STEMMER

The Tigrigna stemming algorithm is implemented as a sequential program using Python programming language. The algorithm is implemented in longest match approach and Affixes are removed through the process of matching the input word to the list of affixes in the rules. The algorithm uses rules in removing single and concatenated affixes. It removes the affixes without iteration. The researcher has collected the affixes used in Tigrigna to form different word variants. Using this finite number of affixes all possible combinations of the affixes are created and the correct ones are selected to form the rules. The prefix removal step has 10 general rules based on the groups of all the possible prefixes and there are more rules within these general rules used to handle exceptions. The suffix removal step also has 7 general rules and other specific rules to handle exceptions.

To illustrate how the algorithm works the following examples are used. The following words*‘ztegebere’,‘zeymgebere’,‘ztetegebere’,‘kemztegebere’,‘slezeyegebere’,‘`Intzeytege bere ‘`IndHrzeyegebre’,‘kemzeytetegebere’* are variants of the root word *‘gbr’* and the variations of these words is due to prefixes *(‘zte-’, ‘zeym-’, ‘ztete-’, ‘kemzte-’, ‘slezeye-’, ‘`Intzeyte-’, ‘`IndHrzeye-’, ‘kemzeytete-’)* attached to the word. For example, the word *‘kemzeytetegbere’* is first checked if it appears in the stopword list and since this word is not in the stopword list it passes to the next step. Next the number of radicals appearing in the word is counted. The word *‘kemzeytetegbere’* has 9 radicals which is above 3, the minimum required radicals for a word to be stemmed, and then the word passes to the next step. The 10-letter prefix ‘*kemzeytete-*’ is removed and the word ‘*gbere’* is returned as a result. The same process is executed for all the other words with prefixes.

If a single word contains a possible combination of prefixes with common starting substring, the algorithm only removes the longest possible prefix to avoid errors in stemming the word. For example the word ‘kemzeyteteHadege’ has prefixes with common starting substring such as ‘kem-’, ‘kemz-’, ‘kemze-’, ‘kemzey’, ‘kemzeyt’, ‘kemzeyte’, ‘kemzeytet’, ‘kemzeytete’.

No.	Prefixes removed	The word to be stemmed	Word after stemming
1	kem-	KemzeyteteHadege	zeyteteHadege
2	Kemz	KemzeyteteHadege	eyteteHadege
3	Kemze	KemzeyteteHadege	yteteHadege
4	Kemzey	KemzeyteteHadege	teteHadege
5	Kemzeyt	KemzeyteteHadege	eteHadege
6	Kemzeyte	KemzeyteteHadege	teHadege
7	Kemzeytet	KemzeyteteHadege	eHadege
8	Kemzeytete	KemzeyteteHadege	Hadege

Table 4. 6:Sample of the prefix removal

The correct stem after proper prefix removal is ‘Hadege’. The above table shows removing the prefixes ‘kemz-’, ‘kemze-’, ‘kemzeyt’, and ‘kemzeytet’ from the word ‘kemzeyteteHadege’ gives the stems ‘eyteteHadege’, ‘yteteHadege’, ‘eteHadege’, and

‘eHadege’ respectively. These stems are incorrect stems and there are no possible prefixes the can be removed from these stems. In order to avoid such results, the stemmer starts from the longest prefix having 11-letters and matches until the appropriate prefix is found.

The suffix removal algorithm follows the same steps as the prefix removal. For example the root word ‘SnH’ have variant forms such as ‘SeniHka’, ‘SeniHkum’, ‘SnHatom’, ‘SeniHkana’, ‘SnHknana’ ‘SeniHkumna’, SnHkumnan’,‘SeniHkumleyn’,‘SnHkumwom’, ‘SeniHkayom’. To stem the word ‘SeniHkumleyn’ is first checked if it appears in the stopword list and since this word is in not in the stopword list it passes to the next step. Next the number of radicals appearing in the word is counted. The word ‘SeniHkumleyn’ has 8 radicals which is above 3, the minimum required radicals for a word to be stemmed, and then the word passes to the next step. The 7-letter suffix ‘kumleyn-’ is removed and the word ‘SeniH’ is returned as a result. The same process is executed for all the other words with suffixes.

No.	Word before stemming	Suffix to be removed	Word after stemming
1	SeniHka	-ka	SeniH
2	SeniHkum	-kum	SeniH
3	SnHatom	-atom	SeniH
4	SeniHkana	-kana	SeniH
5	SnHknana	-knana	SeniH
6	SeniHkumna	-kumna	SeniH
7	SnHkumnan	-kumnan	SeniH
8	SeniHkumleyn	-kumleyn	SeniH
9	SnHkumwom	-kumwom	SeniH
10	SeniHkayom	-kayom	SeniH

Table 4. 7: Sample of the suffix removal

4.8 EXPERIMENTS AND DISCUSSIONS

A series of experiments is 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 Tigrigna corpus created for the development of the stemmer. This corpus covers areas such as politics, economics, religion, medical, sport, and art. The researcher tested the algorithm on a sample of two data sets extracted from the corpus which have 17712 words and contains combined 5437 unique words collected from all categories.

4.8.1 EVALUATION OF THE STEMMER

There are several criteria for judging stemmers: correctness, retrieval effectiveness, and compression performance. There are two ways in which stemming can be incorrect: overstemming and understemming. When a term is overstemmed, too much of it is removed. Overstemming can cause unrelated terms to be conflated. Understemming is the removal of too little of a term. Understemming will prevent related terms from being conflated. The researcher uses the Paice evaluation method to evaluate the quality of the stemmer. In this evaluation method, the quality of the stemmer is assessed by counting the number of identifiable errors during the stemming process. The input words from various samples of texts have to be semantically grouped.

Ideally, a good Tigrigna stemmer will stem all words from the same semantic group to the same stem. But due to the irregularities which are prominent to Tigrigna language the stemmer unavoidably makes mistakes. This is also true for all other natural languages and no stemmer can be expected to work perfectly. A good Tigrigna stemmer should obviously produce as few overstemming and understemming errors as possible. The evaluation of this Tigrigna stemmer is done by counting these errors for the sample of texts. In this evaluation a correctly stemmed word is considered as any word without prefixes and suffixes attached.

4.8.2 THE RESULTS

The Tables 4.8 and 4.9 below shows the summarized results of the evaluation.

Dataset	Words	Correct stems	Percentage
DS1	3122	2659	85.8%
DS2	2315	1998	86.3%

Table 4. 8: Correct Stems

Table 4.9 illustrates the distribution of errors for the incorrectly stemmed words in each data-set.

Dataset	Errors	Overstemming(%)	Understemming(%)
DS1	463	73.4%	26.6%
DS2	317	81.3%	18.7%

Table 4. 9: distribution of errors

From the manual assessment done on the stems 85.8% of the useful words of the first data set were stemmed correctly. 73.4%of the erroneous stems were the result of over-stemming, meaning that the algorithm removed more letters than it should. The rest 26.6% of the words were under-stemmed and the stemmer wasn't able to convert the stem correctly to match the desired one, based on rules of the stemmer. From the manual assessment done on the results of the second data set 86.3 % of the words were stemmed correctly. 81.3% of the errors occur due to overstemming and 18.7% occur due to understemming. Based on the Table 4.9 it can be concluded that overstemming errors are higher than understemming errors. These errors occur because of the large exception words in the Tigrigna language as well as the high inflectional character of the language.

CHAPTER FIVE

CONCLUSIONS AND RECOMMENDATIONS

5.1 CONCLUSIONS

This thesis presented the development of a rule-based stemming algorithm for Tigrigna language. The stemmer is the first rule-based stemmer for Tigrigna language. In Tigrigna there are many exceptions for making stemming rules. The researcher has considered these exceptions in designing the stemmer. Based on the experiments carried out for this study and the results obtained, the following conclusions are presented.

- The results obtained from experiments shows that the algorithm stems the words with an accuracy rate 86.1%.
- Context-sensitive rules are not included in the stemmer because there are no common context-sensitive rules that work commonly for group of words.
- The stemmer does not give similar accuracy on different data sets. The accuracy depends on the words to be stemmed. Some documents contain simple words to stem and other contains complex words to stem, therefore their accuracy may differ depending on the test data set.
- The tests were done on a small collection, so the effect of the stemmer on bigger collection is not known.
- Words that the algorithm fails to analyze are normally foreign words, irregular words or words that do not have trilateral roots.
- Stemming words with single letter suffixes is the most difficult task in the suffix removal process.
- High Errors of overstemming and understemming occur due to morphological complexity of the language.

- One of the biggest difficulties in building this stemming algorithm was that for nearly every rule formulated there are exceptions. Therefore, the algorithm had to use exceptions rules.
- The stemmer is not treating irregular verbs, but they do not seriously affect the results.

5.2 RECOMMENDATIONS

Based on the findings of this study and the knowledge obtained from the literature, the following recommendations are forwarded for future work.

- Researches should be conducted using Tigrigna stemmer on Tigrigna Information Retrieval system to access its impact over recall and precision.
- Evaluation of the Tigrigna stemmer on Tigrigna Information Retrieval system will suggest the best substitution between understemming and overstemming that can be achieved by dropping or adding a few suffixes in the list.
- A more thorough error analysis is required to ascertain what improvement is possible by including iterative rules, and whether such rules will substantially increase the computational cost.
- Moreover, the stemmer has to be tested with large amount of texts to prove its real performance. To succeed in this regard there is a need to apply the Tigrigna stemmer in a web search engine, which retrieves information from Tigrigna texts. Then we can have a complete view of the stemming system and the returned results after every search request.
- 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 Tigrigna language.
- It would be interesting to implement statistical stemmers and see how it performs for Tigrigna language.

Bibliography

Almuhareb, A. Al-Thubaity, M.S. Khorsheed, A. Al-Rajeh, S. Al-Harbi (2006). Automatic Arabic text classification, King Abdulaziz City for Science and Technology, Riyadh, Saudi Arabia.

Adamson G., J. Boreham. (1974) The use of an association measure based on character structure to identify semantically related pairs of words and document titles. Information Processing and Management, vol. 10(7/8), pp. 253-260.

Alemayehu, N., Willett, P. (2002). Stemming of Amharic words for information retrieval. Literary and Linguistic Computing 17(1), 1–17

Alemayehu, N., Willett, P. (2003).The effectiveness of stemming for information retrieval in Amharic. Emerald Research Register 37(4), 254–259 .

Amanuel Sahle (1998). sewasew Tigrigna bsefiḥu. Lawrencevill, NJ, USA: Red Sea Press.

Andrew Spencer(1991). “Morphological Theory: An introduction to word structure in generative grammar”. Oxford: Blackwell Publishers.

Argaw, A.A., Asker, L. (2007b). An Amharic stemmer: Reducing words to their citation forms. In: Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, pp. 104–110. ACL, Prague, Czech Republic Workshop on Computational Approaches to Semitic Languages

C.D. Paice (1996). Method for Evaluation of Stemming Algorithms Based on Error Counting, Journal of The American Society for Information Science 47 no. 8, 632_649.

C.J. van Rijsbergen, S.E. Robertson and M.F. Porter (1980) “New models in probabilistic information retrieval”, British Library Research and Development Report, no.5587.

Daniel Teklu(2008). “Zebenawi sewasw quanqua Tigrigna”, Mekelle: Mega printing

enterprise,.

Dawson J. (1974) Suffix removal for word conflation. In Bulletin of the Association for Literary & Linguistic Computing. Vol. 2(3), pp. 33-46.

Debela Tesfaye, Ermias Abebe. Designing a Rule Based Stemmer for Afaan Oromo Text. International Journal of Computational Linguistics (IJCL), Volume (1): Issue (2), October 2010.

Fox, C. (1990) A stop list for general text. SIGIR Forum, 24, 19-35.

Frakes W.B. (1984) “Term conflation for information retrieval”. Proceedings of the 7th annual international ACM SIGIR conference on Research and development in information retrieval, 383-389

Frakes W., R. Baeza-Yates (1992). Information Retrieval: Data Structures & Algorithms. Englewood Cliffs, NJ: Prentice-Hall.

Girma Berhe (2001). “A Stemming algorithm development for Tigrigna language text documents”, Master Thesis, Fuculity of Informatics, Department of Information Science,

Goweder A. and De Roeck, A. (2001). Assessment of a significant Arabic corpus. Presented at the Arabic NLP Workshop at ACL/EACL, Toulouse, France,.

Gregory T.Stump (2001) Inflectional and Derivational morphology, Kentucky university,USA: Cambridge University Press.

Hafer, M., and S. Weiss (1974). "Word Segmentation by Letter Successor Vaieties," Information Storage and Retrieval, 10, 371-85..

Harman, D. (1991) How effective is suffixing? In Journal of The American Society of Information Science. Vol. 42, No 1. pp. 7-15.

Hui, B.(1998).The Role of Morphology in Machine Translation. Department of Computer Science at the University of Water loo.

Hull, D.A. (1996). Stemming Algorithms: A Case Study for Detailed Evaluation. Journal of the American Society for Information Science, 47(1), 70-84.

J.B.Lovins (1968). “Development of a stemming algorithm”, Mechanical Translation and Computational Linguistics 11, pp. 22-31.

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

Kassa G., Daniel G. Siwasw Tigrigna.Addis Ababa:Mega printing enterprise,2004.

K. Darwish and D. Oard. 2002. CLIR Experiments at Maryland for TREC-2002: Evidence combination for Arabic-English Retrieval. In Proceedings of TREC 2002, Gaithersburg, Maryland.

Khoja, S. and Garside, R. (1999) Stemming Arabic text. Computing Department, Lancaster University, Lancaster.

Krovetz R. Viewing Morphology as an Inference Process. In Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 191-202. ACM. New York. 1993.

Krovetz R.(1995).Word sense disambiguation for large text databases. PhD Thesis. Department of Computer Science, University of Massachusetts Amherst.

Lema Lessa2003. “Development of stemming algorithm for wolaytta text”, Master Thesis Addis Ababa University, Faculty of Informatics, Department of Information Science.

L.S.Larkey and M. E. Connell(2001). Arabic information retrieval at UMass. In Proceedings of TREC 2001, Gaithersburg: NIST.

L.S.Larkey, L. Ballesteros and M.E.Connell. (2002). Improving stemming for Arabic information retrieval: Light Stemming and co-occurrence analysis. In SIGIR 2002, Tampere, Finland: ACM, 2002.

Lennon M.,Pierce D.S.,Tarry B.D. and Willett P. (1981):An evaluation of some conflation algorithms for IR.Journal of information Science,3,pp.177-183.

Lewis, M. Paul (ed.), (2009). Ethnologue: Languages of the World, Sixteenth edition.Dallas, Tex.: SIL International. Online version: http://www.ethnologue.com/

Lovins J. (1968) Development of a stemming algorithm. Mechanical Translation. And Computational Linguistics.11, pp. 22-31.

Markov Models (2003). In Proceedings of Conference on Information and Knowledge Management (CIKM03), pages 131-138, New Orleans, LA. ACM Press.

Mayfield J. and McNamee P. (2003). Single N-gram Stemming. In Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval, pages 415-416, Toronto, Canada, ACM Press.

M. Aljlayl and O. Frieder (2002). On Arabic Search: Improving the retrieval effectiveness via a light stemming approach. In Proceedings of CIKM'02, VA, USA.

M.F. Porter(1980). "An algorithm for suffix stripping", Program, 14(3), pp. 130-137.

M. Popovic and P. Willett (1992). The Effectiveness of Stemming for Natural-Language Access to Slovene Textual Data, Journal of the American Society for Information Science 43.

Paice C. (1990) Another stemmer. In Proc. of SIGIR Forum, vol. 24(3), pp. 56-61.

PopoviE M. & Willett P. (1990). Processing of documents and queries in a Slovene language free text retrieval system. Literaryand Linguistic Computing, 5, 182-190.

Porter M. (1980) An algorithm for suffix stripping. Program 14, 3. pp. 130–137.

R.M.Voigt (1987).The classification of central Semitic. Journal of Semitic Studies,(32):1–21,33

Ricardo Baeza-Yates and Berthier Ribeiro-Neto (1999). Modern Information Retrieval. ACM Press/Addison Wesley.

Ronald M. Kaplan (2005). A Method for Tokenizing Text

Savoy J. (1993). Stemming of French words based on grammatical categories.Journal of the American Society for Information Science, Vol. 44, No 1, pp. 1-10.

The 2007 Population and Housing Census of Ethiopia(2007): Statistical Report for Tigray Region, CSA 2007 National Statistics, Table 2.1.

W. Mekonen(2000). “Development of stemming algorithm for Affan Oromo anguage text”, MSc thesis faculty of informatics, Addis Ababa University, Addis Ababa.

Wightwick, J. and Gaafar, M. (1998) Arabic verbs and essentials of grammar. Chicago: Passport Books.

Xu J., Croft B. (1998) Corpus Based Stemming Using Coocurrence of Word Variants. In ACM Transactions on Information Systems, vol. 16, No 1. pp. 61-81..

Yacob, D. (1997) SERA

Zipf, H.P. (1949). Human Behaviour and the Principle of Least Effort, Addison-Wesley, Cambridge, Massachusetts.

APPENDCES

APPENDIX I: List of stopwords compiled for the stemmer

`ab	`ab	`abti

`ane
`abey
`ayen
`ala
`alo
`abzi
`abay
`abaki
`abaka
`abakn
`abakum
`ab`an
`ab`om
`abana
`ab`atom
`ab`aten
`abzuy
`abzi
`ane
`azyu
`antewo
`abey
`ab`u
`abza
`abzeHa
`alewa
`alewo
`amet
`ametawi
`ayka`alen
`aykonen
b
bSbuQ
bezi
btewesaki
btewesa`ki
b`amhoy
bzey
bzeyka
beti
beta
betom
beten
btom
bten
beti`om
beti`a
beti`an
beti`aten
b`ana
b`aka
b`aki
b`ay
b`akum
b`akn
b`I`om
b`aten
b`an
b`akatkum
b`akatkn
bkulom
bkulna
bkulakum
bkulkn
bebHade
b`kemzi
bzeyka
b`u
bsenki
bmbal
dHri
dHriHzi
dHrit
dHritn
dma
de`a
gn
gna
gena
gna
gda
gegele
gele
gelegele
Hji
Hzi
HalHalifu
HdHd
Hade
Hzi
Hezi
HadeHade
`Intkewn

`Int`kewn
`Iti
`Ita
`Iyu
`Inate
`Ina
`Iyom
`Itom
`Imo
`Izuy
`Iwn
`ina
`Izi
`Iya
`Iyaten
`Iyatom
`Itom
`Iye
`Iyen
`Iwn
`Izuy
`Imber
`Iwn
`Int`kewn
`Intekonewn
`Intekonegn
`Intekonegna
`Inteykones
`Iski
`Imber
`ilom
`Imbi
`Isi
`Indegena
`ina
`ilom
`ilu
`ilna
`ilen
`ilkn
`ilkum
`ile
`ilaten
`ilatkum
`Intelo
`Intela
`Inteleku
`Inteleki
`Intelekum
`Intelew
`Intelekn
`Intelena
`Intelekatkum
`Intbahal
`Intekoynu
`Intekoyne
`Intezeykone
`Intekoyna
`Intekonkum
`Intekoynom
`Intekoynka
`Intekoynki
`Intekoynen
`Intekoynna
`Intay
`Inkelena
`Itom
`Iti
`Ita
`Iti`a
`Iti`om
`Iti`atom
`Iti`aten
`Iti`an
`Ituy
`Izuy
`Izi`a
`Izom
`Izi`aten
`Izi`atom
`Izi
`Izi`a
`Izi`om
`Izi`an
`Intaway
`Intaweyti
`Intawot
`Ikele
`Ikelit
`Igele
`Igelit
`Int`konu
`Izi yu

`Intaynet
`Inted`a
`ane
`Iwe
`Inko
`Int`kona
`Iwn
`Iskab
`ilka
`Intezeykone
kab
kali`I
kndey
kndi
kem
kekem
kbl
kulu
kemti
kemnatom
kemnata
kemzeyblna
kemzeyblom
kemzeybley
kemzeyblki
kemzeyblka
kemzeyblkn
kemzeyblkum
kemzeybla
kemzeyblu
kemzeyblen
ke`a
kabzi
kemza
kemti
kemten
kemtom
kemta
kemti`om
kemti`an
keto
ksab
kndi
ksy
kulom
kulna
kulkn
kulatna
kulatom
kulukum
kulekatkn
kulaten
kulu
kemey
kali`I
kal`ot
kemzi
kem`u
kemay
kem`a
kemana
kemakum
kemakn
kem`om
kemaka
kemaki
kem`atom
kem`aten
ksto
kstay
kemzi`om
kabzi`om
kemzi`an
kemz`konu
kem`an
kemana
kemaka
kemaki
kemzuy
kemzi`a
kemzi`atom
kem`u
kulkn
kabta
ke`a
kone
ke`a
kabtom
kndey
kWa
kndey
kulom
kndti

kem`uwn
kemz`kone
`kone
kone
koynu
koyna
koynom
koyne
kezi
`kezi
kemzelewa
kemzelewu
kemzelena
kemzelekum
kemzelekn
kemzela
kemzeleku
kemzeleka
kemzeleki
kemzelewen
kelena
k`kewn
l`Ili
ms
malet
maleta
maletu
maletey
maletki
maletkum
maleten
maletkn
maletna
maletom
maletka
me`az
men
msay
msaki
msaka
msakum
msana
ms`om
ms`atom
ms`an
msakn
msakatkn
msmen
ms`a
ms`u
mstom
msta
msti
msten
m`Intizi
msti
mszi
ms
manm
msms
mntaysi
me`az
nkule
nKulu
n`kulu
nKulom
nKulen
nKulna
nKulatna
nabti
nayti
neti
nay
netom
nab
netuy
neta
netom
neti
nezi
nayti
natu
natey
nata
naten
natna
natom
nejew
natatен
natatom
natki
natka
natkum

natatkum
natateN
natkN
nska
nsu
nsa
nsom
nHna
nsatom
nsateN
nsa
nsu
nska
nski
nHna
nskN
nsatom
nsom
nseN
nsateN
nskum
nmntay
nmeN
naymeN
neti
neta
netom
neteN
nafti
nafta
naftom
naftN
nabay
nabaki
nabana
nabaka
nab`a
nab`u
nab`om
nab`an
nabakN
nabakum
nab`atom
nab`ateN
n`ay  n`aki
n`aka
n`u`u
n`a`a
n`ana
n`I`om
n`ateN
n`akN
n`akum
n`a`atom
n`akatkN
n`e`an
nabzi`a
nabzi`om
nabzi`an
nab`atom
neza
nezi
nezuy
nezi`a
nezi`om
nezi`an
nezi`atom
nezi`ateN
nezeN
nezom
nzom
nzeN
nzelewom
nzelewen
nzelena
nayzi
nayza
nayzom
nayzeN
nabtom
ntom
nteN
natatkN
nenay
nbmlu`om
nenateN
nla`Ilewot
la`Ilewot
sle
slezkone
slezuy
slezi
sleziz`kone

yelen	TraH	zeleki
yelan	taHti	zelena
yelekun	wn	zelekn
yelekan	wezete	zeyblu
y`kun	wetru	zeybla
yelenan	weywun	zeybley
yelewn	wey	zeyblka
yelewan	wun	zeyblki
ykunde`a	weyke	zeyblom
ykunde`amber	wey	zeyblen
ykunmber	wela`IkWa	zeyblna
yelbon	wala	zeyblkum
y`ay	welawun	zeyblkn
qdm	wHudat	zeykones
qdmit	wHud	zyada
qdmi	zelew`ka	z`kone
slezeyelewu	z`kewn	zbelu
slezelo	z`konet	zbela
slezela	zelo	zbelna
slezeyele	zela	zbele
slezeyela	zelewa	zbelkn
slezelena	zelewu	zbelkum
slezelo	zeleka	
SbuQ	zeleku	
Tray	zelekum	

APPENDIX II：List of Tigrigna prefixes compiled for the stemmer

`IndHrzeyte	slezeytete	kemzeyte
`IndHrzeyt	`IndHrzeyt	`Inteyte
kemzeytete	`IndHrzey':	`Intete'

`Indate	`Ite	Key
slezeye	Zete	Beb
slezeyt'	Zeym	`ay
kemzey	Kemt	`In
kem`It	Ztet	zte
kemzte	`Ina	nen
mstete	Keye	nkn
slezte	`ayn	kne
slezey	`ayt	sle
`Inate	`Int	kte
`Intez	ztet	kne
`IndHr	mste	kem
kemte	kndi	zey
kemIn	ktet	ze
keyte	slez	te
`Inte	kndt	ti
zeyte	ksab	`I
kemze	nzte	Kt
ztete	nkey	Ke
ktete	bate	kn
Slete	bzey	nm
Sleze	bebi	nz
`aymte	bzte	nb
`Intey	tey	nk
`Inkey	nze	bz
ksabz	bte	ba
`ayte	nme	bz
`Inte	b`a	bb
`Inda	bmt	bm
Kemz	ayn	ms
Slet	`It	mt

z

k

n

i

o

u

e

APPENDIX III: List of Tigrigna suffixes compiled for the stemmer

teNatat	klomn	nalka
kumleyn	Natat	kulka
knaleyn	titat	knala
aynetom	winet	Kiyen
knalom	nakum	Kayen
kumwom	atkum	Nayom
Kumwom	alkum	tatat
tatnan	kleyn	atom
tatomn	nayen	tatn
kalomn	Kaley	otat
nayomn	atnan	klom
awinet	naley	tlom
waynet	wetay	lomn
netawi	tatna	knom
tatkum	knana	ttat
nalkum	Knana	ynet
kulkum	Kumna	tkum
kumley	kumna	ukum
kaleyn	Knalu	uwon
knaley	ynetn	uwom
knalna	Kumni	kley
kumlen	kumni	mley
knalen	Knani	tley
kumwen	knani	atay
tatom	nalki	away
kalom	`klom	kana
nalom	Kulki	kana
kayom	nalen	nana
kayom	kalen	tlna

klna
tatu
netu
nalu
kalu
netn
kani
kani
nani
omni
umni
atni
atna
naki
tlki
klen
tlen
atkn
naka
tlka
kila
nala
kula
kyen
aten
uwen
knen
knen
elom
tat
uwo

atn
atu
tom
lom
wom
yom
omn
net
awi
tla
Kum
Kum
kWa
ley
tay
una
kna
tna
kna
yen
mlu
tlu
klu
knu
nan
uni
kni
len
ukn
om
Ka

ka
en
wo
at
ot
on
et
Ki
ki
na
Kn
tn
Ku
ey
ay
na
ku
tn
kn
u
i
a
n

APPENDIX IV: Transliteration Ethiopic script to Latin equivalent

ሀ	he
ለ	le
ሐ	He
መ	me
ሠ	se
ረ	re
ሰ	se
ሸ	xe
ቀ	qe
በ	be
ቨ	ve
ተ	te
ቸ	ce
ኀ	he
ነ	ne
ኘ	Ne
አ	ʻa
ከ	ke
ኸ	ʻke
ወ	we
ዐ	ʻe
ዘ	ze
ዠ	Ze
የ	ye
ደ	de
ጀ	je
ገ	ge
ጠ	Te
ጨ	Ce
ጰ	Pe
ጸ	Se
ፀ	Se
ፈ	fe
ፐ	pe
ሁ	hu
ሉ	lu
ሑ	Hu
ሙ	mu
ሡ	su
ሩ	ru
ሱ	su
ሹ	xu
ቁ	qu
ቡ	bu

ቩ	vu
ቱ	tu
ቹ	cu
ኍ	hu
ኑ	nu
ኙ	Nu
ኡ	ʻu
ኩ	ku
ኹ	ʻku
ዉ	wu
ዑ	ʻu
ዙ	zu
ዥ	Zu
ዩ	yu
ዱ	du
ጁ	ju
ጉ	gu
ጡ	Tu
ጩ	Cu
ጱ	Pu
ጹ	Su
ፁ	Su
ፉ	fu
ፕ	pu
ሂ	hi
ሊ	li
ሒ	Hi
ሚ	mi
ሢ	si
ሪ	ri
ሲ	si
ሺ	xi
ቂ	qi
ቢ	bi
ቪ	vi
ቲ	ti
ቺ	ci
ኂ	hi
ኒ	ni
ኚ	Ni
ኢ	ʻi
ኪ	ki
ኺ	ʻki
ዊ	wi
ዒ	ʻi
ዚ	zi
ዢ	Zi

ዩ	yi
ዲ	di
ጂ	ji
ጊ	gi
ጢ	Ti
ጪ	Ci
ጲ	Pi
ጺ	Si
ፂ	Si
ፊ	fi
ፒ	pi
ሃ	ha
ላ	la
ሓ	Ha
ማ	ma
ሣ	sa
ራ	ra
ሳ	sa
ሻ	xa
ቃ	qa
ባ	ba
ቫ	va
ታ	ta
ቻ	ca
ኃ	ha
ና	na
ኛ	Na
አ	ʻa
ካ	ka
ኻ	ʻka
ዋ	wa
ዓ	ʻa
ዛ	za
ዣ	Za
ያ	ya
ዳ	da
ጃ	ja
ጋ	ga
ጣ	Ta
ጫ	Ca
ጳ	Pa
ጻ	Sa
ፃ	Sa
ፋ	fa
ፓ	pa
ሄ	hE
ሌ	lE

ሔ	HE
ሜ	mE
ሤ	sE
ሬ	rE
ሴ	sE
ሼ	xE
ቄ	qE
ቤ	bE
ቬ	vE
ቴ	tE
ቼ	cE
ኄ	hE
ኔ	nE
ኜ	NE
ኤ	ʻE
ኬ	kE
ኼ	ʻkE
ዌ	wE
ዔ	ʻE
ዜ	zE
ዤ	ZE
ዬ	yE
ዴ	dE
ጄ	jE
ጌ	gE
ጤ	TE
ጬ	CE
ጴ	PE
ጼ	SE
ፄ	SE
ፌ	fE
ፔ	pE
ህ	h
ል	l
ሕ	H
ም	m
ሥ	s
ር	r
ስ	s
ሽ	x
ቅ	q
ብ	b
ቭ	v
ት	t
ች	c
ኅ	h
ን	n
ኝ	N
አ	ʻl
ከ	k
ኽ	ʻk
ው	w
ዕ	ʻl
ዝ	z
ዥ	Z
ይ	y
ድ	d
ጅ	j
ግ	g
ጥ	T
ጭ	C
ጵ	P
ጽ	S
ፅ	S
ፍ	f
ፕ	p
ሆ	ho
ሎ	lo
ሐ	Ho
ሞ	mo
ሦ	so
ሮ	ro
ሶ	so
ሾ	xo
ቆ	qo
ቦ	bo
ቮ	vo
ቶ	to
ቾ	co
ኆ	ʻho
ኖ	no
ኞ	No
አ	ʻo
ኮ	ko
ኾ	ʻko
ዎ	wo
ዖ	ʻo
ዞ	zo
ዦ	Zo
ዮ	yo
ዶ	do
ጆ	jo
ጎ	go
ጦ	To
ጮ	Co
ጶ	Po
ጾ	So
ፆ	So
ፎ	fo
ፖ	po
ሏ	lWa
ሗ	HWa
ሟ	mWa
ሧ	sWa
ሯ	rWa
ሷ	sWa
ሿ	xWa
ቈ	qWe
ቧ	bWa
ቯ	vWa
ቷ	tWa
ቿ	cWa
ኈ	hWe
ኗ	nWa
ኟ	NWa
ኰ	kWe
ዟ	zWa
ዧ	ZWa
ዷ	dWa
ጇ	jWa
ጐ	gWe
ጧ	TWa
ጯ	CWa
ጷ	PWa
ፏ	SWa
ፏ	fWa
ፗ	pWa
ቍ	qWu
ኍ	hWu
ኵ	kWu
ጕ	gWu
ቌ	qWi
ኌ	hWi
ኩ	kWi
ጒ	gWi
ቋ	qWa
ኋ	hWa
ኳ	kWa
ጓ	gWa
ቈ	qWE
ኈ	hWE
ኴ	kWE
ጔ	gWE
ኧ	ea
ቐ	Qe
ቑ	Qu
ቒ	Qi
ቓ	Qa
ቔ	QE
ቕ	Q
ቖ	Qo
ቒ	Qwa

APPENDIX V: comparison of Tigrigna words before stem and after stem

Tigrigna words before stemming

tlmi medebat fEderexn mena`Isey tgray Hdar meQele medebat fEderexn mena`Isey tgray me`Itewi `ab tari`k bsenki zneberu sur zsededu tederarebti ba`Idawi megza`Itat mena`Isey bneSa dlyetom tewedibom bahgitatom zenSebarQlu kunetat bzeymnbaru `abti `Iwan `ab mengWe bihErat zneberu zeyftHawi `Iblela mena`Isey bflay dma mena`Isey tgray bwlqen bwdaben neti megza`Iti kab suru nmmHaw bztefelaleye melk`u zgleSu teQawemtat `Inageberu meSi`om `Iyom bmeseret `Izi Hto lm`atn bhErawi ma`Irnetn Trzi bmbSHu nay Hafax teQawumo nab ma`Ibel telewiTu bwHudat deqi hzbi ztejemere bretawi tegadlo dHri merir meswa`Iti hzbitat bqWanqo`om kTqemu ftHawi lm`at kregageS nay bihErebhEresebat hzbtatn nay Habar TQmi zeregagS Hge-mengsti kQreS mena`Isey y`kun kal`ot kflitat dlEtatom zgelSlomn meselatom zeregagSlom tewedibom nay lm`at Haylitat koynom n`knqesaqesu mcwi kunetat tefeTiru `Iyu slez`kone dma nezi bmeswa`Iti ztefeTere TuTuH bayta nmTqam mena`Isey tgray bztefelaleyu maHberat tewedibom tesateftin terebaHtn nayti lm`at `Ina`konu yrkebu bmeseret `Izi `ab zHalefu `ametat `ab tgray zrkeba ztefelaleya wdabetat mena`Isey nab Hade Slal bmmSa`I `ab zurya mena`Isey zsrHu lm`at bzHaxe nmmraH fEderexn mena`Isey tgray `ab kab Tabya jemiru ksab kll teTayxu bezi dma bebi derej`u `amerarHatat kQomu tegeyru `Iyu mebegesi gemgam `afeSaSma medebat `aTaQalali `aQmi fEderexn mena`Isey mTn`kar bzmlket `ab zHalefe `amet `Iti wdabe nmejemerya zTayexn zwdebn slezelo TraH `Inteykones gena mTn`kar zedlyu mu nu bm`Iman `Iti `agedasn qulfin medeb geyrna zwesednayo `aQmi `Iti fEderexn nm`Ibay zsrHu sraHti `Iyom neyrom `Inte `kone gn `Izi bm`knyat nay bejet HSretn `ab bebiderej`u zelewu `amerarHa nayti fEderexn tewefiyu nab CbuT tegbar bzeym`Itawun ztefeSeme `aykonen bm`k nyat `Izi dma `ab bfEderexn mena`Isey tgray ksrHu zteweTenu medebat keytefeSemu zterefu `Int`konu b`anSar `Iti zneberu medeb nayzemfSam xgr kemzs`Ib bebiderej`u mr`ay yke`al bkll dereja zelo `amerarHa `Iti fEderexn `awunta `Iti fEderexn bflay mejemeriya `Intmsret dHar wun `Inte`kone mebegesi tlmitat bmwSa`I nab meHawurat `ab ml`a`k b`awenta zgleSu jmarotat tera`Iyom `Iyom `ab mereSa znebere `ab m`Iwat ztewadede `afeSaSma mnbaru kremtawi wefri temharo yuniversiti `ab mfSam znebere mt`Issar SbuQ temokro ztere`kebelu m nu `aluta la`Ilewot `amerarHa nayti fEderexn tenabibkan teredadi`I`kan nab CbuT medeb zeym`Itawu

`abti zweSe tlmi nkfSem zgber nay degefn kttln mn`as kab kal`ot kllat temokro SemiQ`ka nab nay kllna wdabe ksefH nayzeymgbar nezi zedli bejet zeymt`Illax kabtom zneberu xgrat kozneQelu koynom bmeseret `Izi ke`a `ab bfEderexn mena`Isey tgray nkfSemu tetaHizom keytefeSemu zterefu medebat kemzs`Ib mr`ay yke`al bebi derej`u zrkebu `amerarHatat `Iti fEderExn gnzabe zCbTlu medre`kat mdlaw bzmlket `abal wdabetat `ab `aserarHa `awedadban l`u`kn kal`ot SHufatn `Iti fEderExn bzmlket glSnet zfeTrelu slTenatat nmhab ztetHaze medeb kndti zdle zeykede m nu lojstikawun faynansawun `aQmi `Iti fEderexn m`Ibay medebat `Iti keytefeSmu terifom `Iyom naytu fEderExn Hgawi sebnet nmwSa`I tedegagami Sa`Iri kWa `Intetegebere `ab biro ftHi nezi zete`anagd Hgi bzeymhlaw m`knyat ksa`ka`I `ayke`alen nezi Hagazi k`kewn z`k`Il memesreti qale `a`kEba kneQrb teHatitna wun `ab `id SeHafit `Iti fEderExn slezrkeb b`Iwanu keQrb zeym`k`alu bdmr bezom l`Il `ilom zteTeQesu m`knyat `Iti fEderExn nay ba`Ilu Hgawi maHtemn `arman zeymhlaw n`Ingebrom rkbat deritwom mhlawu teTeqesti `Iyom `ab weredan Tabyan zrkebu `amerarHa `awenta kab kll nzweredu medebat `Iti wdabe nmsraH jmarotat mhlaw `aluta kab kll nzweredu medebat meseret bmgbar CbuT kebabiyawi kunetat meseret zgebere tlmitat bmdlaw nmfSam zgber Sa`Iri n`us m nu bflay `ab wereda zrkebu `amerarHana neti ztewehabom Halafnet meseret bmgbar ztefelaleyu nay mena`Isey medre`kat bmfTar `ab kndi msraH neti ztewehabom Halafnet b`kali`I melk`u zSbeyu wHudat `aykonun fEderexn bzmlket zelo nay k`Ilet gnzaben HSretat zeytefetHu xgrat mhlawu zelo Tenkaran d`kumn gontat `ab `Iten `abalat fEderexn wdabetat ba`Ilen `awenta bflay `Iten bdereja kll ztewedeba `abalat `Iti fEderexn nba`Ilten kab gzE nab gzE `InateTena`kera ymeSa mhlawen neti fEderexn nmTn`kar ztemecacewe beri z`kft kab kll ksab taHti Tabya zteTena`kere wdaben meHawrn mhlaw `aluta `Iten wdabetat neti fEderexn meTena`keri zw`Il nay `abalnet mewaCo zeymjmaren `ab medargti `akalat znebere kunetat `awenta mejemeriya `ab mTyaxn `Iti guba`E `ab bebi derej`u zkeyed guba`Etat bbajet nay mtHbbar degefat mhab neyru bQeTta neti fEderexn wala `ay`kun neten `abalat fEderexn wdabetat nay gemgamawi slTanetatn kal`ot nay `aQmi me`Ibey degefat mhab `aluta kem wdabe mena`Isey `Iti fEderexn nmTn`karn `arsu k`ilu TeTewu n`kbln zgberu faynansawi materiyalawun bHayli sebn bmTn`kar melk`u Hagezat nay zeymhab zhlwna medebat qulfi medeb fEderexn mena`Isey tgray b`aQmi mTn`kar fEderexn mena`Isey tgray bmTn`kar serawit lm`at mena`Isey nmhnaS zek`Il ztemecacewe bayta mfTar `agedasn weQtawun bm nu `abzi `amet `Izi zhlwena Qulfi medeb `ab zurya mlax `aQmtat `Iti fEderexn mTn`kar `Iyu `Ilama nhnSet lm`atawi serawitn mtgbar nay `amet `Ibyetn sggrn tlmi

nay mfSam `aQmi zelewen wdabetat mena`Isey tgray mfTar medebat fEderexn mena`Isey tgray nmTn`kar zw`Il ksab br zgmet faynansawi nwatawin nay Hayli sebn hafti mt`I`k`kab tegbarat bebiderej`u zelewu `abalat fEderexn mena`Isey tgray zelewa wdabetat mena`Isey tlmi bayta zegenazebe bebiderej`u kewS`a mgbar feSemti `aQmtat nayti fEderexn bmelk`I serawit nab lm`at z`atwalu mengedi mdlaw zweSe `ametawi medebn `afeSaSma `anfetn bebiderej`u zelewu `amerarHa medeb gnzabe meCebeTi medre`k mdlaw `ab bebiderej`u zkayedu nay mena`Isey medre`kat mena`Isey tgray b`ateHasasban b`are`a`Iyan lm`atawi koynu malet wun beti mengsti qeriSwo zelo nay mena`Isey lm`at pakEj bzgba`I teredi`u tesatafin teTeqamin nk`kewn msraH `ab keteman geSern zelewu mena`Isey `abten qulfi maHberawi, quTebawin poletikawi mnqsqas `abti `atyom tesatefti k`konu msraH `abeyti medebat ml`I`aln mwudabn kremtawi wefri mena`Isey `ab zHalefu `ametat mena`Isey tgray bwlqe y`kun bwdabe bflay `ab `Iwan kremti b`arse tebegso btitoriyal klas y`kun b`kal`ot gulbetawi degefat bmhab zteSegemu mena`Iseyn `aregawiyann `ab mdgaf `abeyti zetebab`u sraHti `InateserHu meSi`om `Iyom btemesasali mengedi `abzi `amet `Izi wun `Inte`kone `ab kremtn kab kremti weSa`i `afti kebabi bzSenHu mena`Iseyn kab yuniversitat n`Irefti nzmeSun nay gulbetn k`Iletn degefat zhblu kunetat nmsraH kem `aby tlmi tetaHizu `Iyu `Ilama kebabiyawi hafti tefeTron Sryetn mHlaw `ab bebi kebabi`u zrkebu zteSegemu mena`Iseyn `aregawuyann bgulbet bflTetn bgenzebn Hagez bmhab maHberawi wHsna zre`kblu kunetat mm`Irray mena`Isey kab yuniversitat n`Irefti zmeSun `afti `kebabi bzSenHu mena`Isey glgalot senay fQad ksatefu bmgbar `ab bebi kebabi`u nHmumat nzteSegemu `aregawuyann mena`Iseyn `ab bebiSHfetu moyawi degef `ab mhabn `ab `abyate tmhrti nay titoriyal degefatn glgalot nay mhab sraHti teTena`kirom kQSlu `Iyom tegbarat nay mena`Isey wdabetatn maHberatn `abalatn kab `abalaten weSa`In zelewu mena`Isey nay meredad`i medre`kat bmkyad `ab tlmen kekatt`o mgbar tegbar `ab bebikebabi`u bneSa glgalot mena`Isey ksraH zgb`o feli`ka mHaz tegbar kab yuniversitat n`Irefti zmlesun neberti `Iti `kebabn nenay ba`Iltom medre`kat bmdlaw `Intaynet neSa glgalot mena`Iseyn kal`otn `aflTo ze`Ibylun zedli mrdda`In mkyad `abeyti medebat guba`E fEderExn mena`Isey tgray mkyad fEderExn mena`Isey tgray mesrati guba`E`u `ab bdereja Tabya weredan klln `Intkayed bmeseret `Iti meteHadaderi denbi `Iti wdabe bebi`ametu guba`E kkayed wesinu `Iyu slez`kone dma `ab bebiderej`u guba`Etat kkayed `Iyu `Ilama `ab bebiderej`u meHawr `Iti fEderExn bmeseret meteHadaderi denbi guba`Etat bmkyad nzHalefu gu`Izotatu bmgmgam nay qeSalay `anfetatn `amerarHatatn zsymelu bayta mmccaw xto bdereja Tabya weradan klln tesatefti `Iti guba`E

Tigrigna text after stemming (stopword excluded)

tlm medeb fEderex mena`Is tgr Hdar meQele medeb fEderex mena`Is tgr me`Itew tari`k zneber sur zseded derareb ba`Id megza`I mena`Is bneS dly wedib hgi nSebarQl kunet mnbar mengWe bihEr zneber ftH blel mena`Is bfl mena`Is tgr bwlq bwdab megza`It sur mHaw felaleye melk` zgleS Qawem geber meSi`o seret Hto lm`at bhEr ma`Ir Trz bSH Hafax Qawumo ma`Ibel lewiT bwHud deqi hzb jemere bret gadlo merir meswa`It hzbi bqWanqo` kTqem ftH lm`at kregageS bihErebhEreseb hzb Habar TQm regagS Hge mengst kQreS mena`Is kfli dlEt zgelS mesel regagS wedib lm`at Hayli qesaqes mcw kunet feTir swa`It feTere TuTuH Tqam mena`Is tgr felaley maHber wedib satef rebaH lm`at `konu yrkeb seret zHalef `amet tgr zrkeb felaley wdabe mena`Is Slal mSa`I zury mena`Is zsrH lm`at Haxe mraH fEderex mena`Is tgr Taby jemir kll Tayx derej` `amerarHa kQom geyr mebeges gemgam `afeSaSm medeb `aTaQalal `aQm fEderex mena`Is mTn`kar mlk zHalefe wdabe jemery zTayex zwdeb mTn`kar dly `Iman `agedas qulfi medeb geyr zwesed `aQm fEderex `Ibay zsrH sraH neyr `kny bejet HSre derej` `amerarH fEderex wefiy CbuT gbar `Itaw feSeme bfEderex mena`Is tgr ksrH weTen medeb feSem teref nSar zneber medeb fSam xgr s`Ib derej` mr`ay yke`al bkll derej `amerarH fEderex `awunt fEderex bfl mejemeriy msr dHar mebeges tlm wSa`I meHawur ml`a`k went zgleS jmar ra`Iy mereS znebere m`Iw wadede `afeSaSm mnbar kremt wefr mharo yuniversit mfSam znebere `Issar mokro re`kebe `alut `amerarH fEderex nabib redadi`I` CbuT medeb `Itaw zweSe tlm fSem zgber degef mn`as kll mokro SemiQ` kll wdabe ksefH mgbar bejet `Illax zneber xgr seret bfEderex mena`Is tgr fSem taHiz feSem teref medeb s`Ib mr`ay yke`al derej` zrkeb `amerarHa fEderEx gnzabe zCbT medre`k mdlaw mlk `abal wdabe `aserarH `awedadb l`u`k SHuf fEderEx mlk glS zfeTre slTena tHaze medeb zdle ykede lojstik faynans `aQm fEderex m`Ib medeb feSm rifom fEderEx Hgaw sebn wSa`I degagam Sa`Ir gebere biro ftH `anagd Hgi mhlaw m`kny ksa`ka`I ke`ale Hagaz z`k`Il memesret qale `a`kEb Qrb Hatit SeHafit fEderEx rkeb b`Iwan Qrb `k`al bdmr l`Il TeQes m`kny fEderEx ba`Il Hgaw maHtem `arma hlaw gebr rkb derit mhlaw Teqes wereda Tabya zrkeb `amerarH `awent kll wered medeb wdabe sraH jmar mhlaw `alut kll wered medeb meser gbar CbuT babiy kunet meser zgebere tlm dlaw fSam zgber Sa`Ir n`us bfl wered zrkeb `amerarHa wehab Halaf meser gbar felaley mena`Is medre`k fTar wehab Halaf b`kali`I melk` zSbey fEderex mlk k`Il gnzab HSre fetH xgr mhlaw

Tenkara d`kum gont `abal fEderex wdabe `awent bfl bderej kll wedeb `abal fEderex gzE gzE Tena`ker ymeS mhlaw fEderex Tn`kar mecacewe beri z`kft kll Taby Tena`kere wdab meHawr mhlaw `alut wdabe fEderex meTena`ker zw`Il `abal mewaCo jmar medargt `akal znebere kunet `awent mejemeriy mTyax guba`E derej` zkeyed guba`E Hbbar degef mhab neyr bQeTt fEderex `abal fEderex wdabe gemgam slTane `aQm me`Ib degef mhab `alut wdabe mena`Is fEderex Tn`kar `ars k`il TeTew n`kbl zgber faynans materiyal bHayl sebn Tn`kar melk` Hagez zhlw medeb qulf medeb fEderex mena`Is tgr mTn`kar fEderex mena`Is tgr Tn`kar serawit lm`at mena`Is hnaS k`Il mecacewe mfTar `agedas weQt zhlw Qulf medeb zury mlax `aQm fEderex mTn`kar `Ilam nhnS lm`at seraw gbar byet sggr tlm mfSam `aQm wdabe mena`Is tgr mfTar medeb fEderex mena`Is tgr Tn`kar zw`Il br zgm faynans nwat Hayl sebn haft `I`k`kab gbar derej` `abal fEderex mena`Is tgr wdabe mena`Is tlm genazebe derej` wS`a mgbar feSemt `aQm fEderex elk`I serawit lm`at z`atwal menged mdlaw zweSe medeb `afeSaSm `anfe derej` `amerarH medeb gnzabe meCebeT medre`k mdlaw derej` zkayed mena`Is medre`k mena`Is tgr teHasasba re`a`Iya lm`at mengst qeriS mena`Is lm`at pakEj gba`I teredi` satafi Teqami teman geSer mena`Is qulf maHber quTeb poletik mnqsqas `aty sateft `abeyt medeb ml`I`al mwudab kremt wefr mena`Is zHalef mena`Is tgr bwlqe bwdabe bfl kremt begso btitoriyal klas b`kal gulb degef Segem mena`Isey `aregawi mdgaf `abeyt tebab` sraHt serH meSi`om mesasal menged krem kremt weSa` kebab SenH mena`Isey yuniversi n`Ireft meSun gulbe k`Ile degef zhbl kunet sraH `aby tlm taHiz `Ilam babiy haft feTr Sry mHlaw kebabi` zrkeb Segem mena`Isey `aregawu bgulb bflTe bgenzeb Hagez maHber wHs zre`kbl kunet mm`Irr mena`Is yuniversi n`Ireft zmeSu `kebab SenH mena`Is glgal senay fQad ksatef gbar kebabi` nHmum Segem `aregawuyan mena`Isey SHfet moyawi degef mhab `abyate tmhrt toriyal degef glgal mhab sraHt Tena`kir kQSl gbar mena`Is wdabe maHber `abal `abal weSa`I mena`Is meredad` medre`k kyad tlm katt` mgbar gbar kebabi` bneS glgal mena`Is ksraH zgb` feli`k mHaz gbar yuniversi n`Ireft zmles nebert `kebab medre`k dlaw neSa glgal mena`Isey `aflT `Iby zedli mrdda`I mkyad `abeyt medeb guba`E fEderEx mena`Is tgr mkyad fEderEx mena`Is tgr mesrat guba`E` bderej Taby wereda kll kayed seret meteHadader denb wdabe guba`E kkayed wesin derej` guba`E kkayed `Ilam derej` meHawr fEderEx seret meteHadader denb guba`E kyad Halef gu`Izo gmgam qeSal `anfet `amerarHa zsymel mmccaw xto bderej Taby werada kll sateft guba`E senadawit komitE wsen meser guba`E mkyad gbar kll wered kayed guba`E