ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATES STUDIES

DEPARTMENT OF INFORMATION SCIENCE

DESIGNINIG A STEMMING ALGORITHM FOR SILT'E LANGUAGE

A thesis submitted to School of Graduate Studies of Addis Ababa University in partial fulfillment of the Requirement for the Degree of Master of Science in Information Science

By

MUZEYN KEDIR ABEDO

June, 2012

Addis Ababa

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATES STUDIES

DEPARTMENT OF INFORMATION SCIENCE

DESIGNINIG A STEMMING ALGORITHM FOR SILT'E LANGUAGE

A thesis submitted to School of Graduate Studies of Addis Ababa University in partial fulfillment of the Requirement for the Degree of Master of Science in Information Science

By

MUZEYN KEDIR ABEDO

Names and signature of Members of Examining Board

Name	**Title**	**Signature**	**Date**
_____	Chairperson	_____	_____
_____	Advisor	_____	_____
_____	Examiner	_____	_____

ACKNOWLEDGMENTS

Allahamdulilah, for everything Allah has made for me and again allahamdulilah.

Here, I would like to take this opportynity to express my most gratitude to those that had helped and inspied me in the completion of this thesis. I owe a great debt of gratitude to my advisor Marta Yifru (PhD). This thesis would not have been possible without her guidance, as well as inspiring and enlightening ideas, comments and suggestions. Thank you very much.

I would like to express my heartfelt thank to Million M. (PhD) for his comment and constructive ideas and I am proud of being his student.

I would like to express my gratitude to my beloved family for their encouragement and prayer ("Dua").

To my frends, thanks for all the moral supports, especially Abubeker Usman, Enyew Ahmedine Nuredin Nasser, Kedir Mohamed, Dr. Elias Gebru and Mohammedamin Nassir for that they have made some important resources available and for their constractive comments and many other individuals that are unable to be listed who have directly or indirectly help me in completion of my thesis.

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

Muzeyn Kedir Abedo

June, 2012

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDECES

LIST OF ABBREVIATIONS AND SYMBOLS

Symbol	Meaning
CV	=Consonant and vowel sequence
‘ ‘	= meaning of words
| **|**	= the English translation of words
1ps	= first person singular
1pp	=first person plural
2sm	=second person singular masculine
2sf	= second person singular feminine
2pm	=second person plural masculine
2pf	= second person plural feminine
3sm	=third person masculine singular
3sf	=third person feminine singular
3pm	=third person masculine plural
3pf	= third person feminine plural

Abstract

Variant word forms that are likely to be encountered in indexing and retrieval are one of the causes of the problems that are involved in the use of freetext retrieval system. The variant word structure s used in indexing and searching are to be expected in determining the relevance of a document to a user query that specifies just a single form. Shrinking the variant words in to one form advances the performance of IR system and this can be achieved by conflation techniques, which is usually stemming that is established in this work. Stemmers are used in information retrieval to reduce as many related words and word forms as possible to a standard form, which can then be used in the retrieval process. This research explores the possibility of developing a stemmer to conflate variant words of Silt'e language.

Silt'e belongs to the Semitic language group. These languages have a common grammatical system based on a root-pattern structure. Consonants bear the basic meaning while vowels form different patterns. Stems are built from consonantal roots before other word forms are built. Silt'e uses affixation and reduplication to derive different word forms from stems. Common affixations are prefix, suffix, and infix. Silt'e uses extensive concatenation of affixes and can result in relatively long words, which often contain an amount of semantic information equivalent to a whole English phrase, clause or sentence. As a result of this complex morphological structure, a single Silt'e word can have very large variants.

To design the stemmer, a sample text was collected from different sources and research paper that explains the morphology of Silt'e language also used and affixes and stopwords collected from this research paper and the sample text document to develope the stemmer. The stemmer, developed in this study is iterative and uses context sensitive and recoding rules that remove prefix, suffix and reduplication of letters (type 1 and type 2). In this experiment the stripping procedure were applied in order: prefix, suffix and finally letter reduplication. The stemmer was tested on a sample data of 1486 words, which were selected randomly from the sample texts. The result of the experiment shows that, the stemmer performs at accuracy of **85.71%**, and brings a dictionary reduction of 34.99% for stem words. Lastly conclution and the possible recommendation for future work were reported.

CHAPTER ONE

1.1 BACKGROUND OF THE STUDY

The activity of archiving written information can be traced back to around 3000 BC, when the Sumerians selected special areas to store clay tablets with cunei form inscriptions. Even then the Sumerians realized that proper organization and access to the archives was critical for efficient use of information. They developed special classifications to identify every tablet and its content.

The need to organize, store and retrieve written information happened to be increasingly important over centuries, particularly with the emergence of paper and printing press. Then computers came in to existence and people understood that it could be used for storing and mechanically retrieving large amounts of information [32].

The collection development problem is aggravated by the growth in available information. In early times, the total available information (knowledge) changed slowly. However, by the year 1800 the amount of scientific publication was doubling every 50 years [33]. More recently with the remarkable growth of science and technology, the rate of increase of available information (knowledge) has greatly accelerated. “Between 1800 and 1966, the number of scientific journals has increased from 100 to 100,000”. At the present time, no upper limit is apparent in the rate of increase of available information items**.**

The dramatic growth in size and variety of available information made the utilization and access of the right information difficult. Therefore, proper organization of information resources matters to get the most out of it and effective and efficient utilization of it**.**

Information retrieval systems (IRs) [45] are designed to facilitate access to stored information. In addition, they are concerned with the representation, storage and organization of information items. The components of an IR system consists of a set of information items, a set of requests and a mechanism to determine which information items are most likely to meet the requirements of the requests. Basically, users of information state their information needs in the form of queries and submit their queries to the information retrieval system. Based on their queries, the system outputs information items that are believed to be relevant to the user query. In other words, the items that have common entries in both the database

and users query are returned to users. This happens when a matching exists between queries and information items. To attain a match between these two components, the items in the documents as well as in the query should be represented in some form.

Frequently, the user specifies a word in a query but variant of this word are present in a relevant document. The variation prevents a perfect match between a query word and word in relevant documents. This problem can be overcome with the substitution of the words by their respective ***stems***. A stem is a portion of a word that is left after the removal of its affixes (i.e., prefixes and suffixes). A typical example of a stem can be the word "comput" which is the stem for the variants "computed," "computing," "computation," and "computations."

Morphology describes how various forms of words are created, and studies structure of words in the language. Suffixing and prefixing are the main and common ways of creating word variations in natural language text [36]. Morphology (the internal structure of words) can be broken down in to two sub classes: inflectional and derivational [6]. Inflectional morphology describes predictable changes of words. These are as a result of syntax, such as changes in person, number, tense and gender. These changes have no effect on a word's part of speech [38] that means; a noun still remains noun after pluralized. For example, kick, kicks, kicking, kicked.

On the other hand, changes of derivational types may affect the word's meaning in part of speech. For example, affix changes from adjective to nouns, from verb to nouns, from noun to verbs, and so on; like friend, friendly, friendliness and friendship.

From either type of morphologies, depending up on the complexity of the morphological natures of language types, very massive variants of words may be resulted from a single word. This large number of occurrences has strong impact on information retrieval systems. Thus, there is a need of automated procedure that can reduce the size of the various words to manageable level, and also capture the strong correlation existing between different word forms [39]. Even if the complexity of morphological variations may differ from one natural language to others, stemming is widely used in information retrieval (IR), with the basic reason that morphological variants represent similar meaning.

Morphological processing becomes widely used for effective and efficient information retrieval [37], machine translation (MT) and text classification [39]. Hence, morphological processing becomes particularly important for information retrieval (IR) because IR needs to determine the appropriate forms of words as index [24].

Information retrieval particularly automatic information retrieval system is an information processing activity which is carried out with the help of automatic equipment. As [1] defined automatic information retrieval system is a software /hardware package that lets different users to access query and retrieve information from the data base.

Stemming is a natural language processing technique that identifies a stem/root from morphologically conflated words using various techniques on inflectional and derivational affixes [36]. It helps to reduce morphological variants of words in to a common form. As an example, 'throws', 'thrower', and 'throwing' are the common variances of a stemmed word 'throw. Therefore, stemming can be applied in different languages to increase searching efficiency and reducing vocabulary size of indexed files in information retrievals [38, 6].

In stemming, all of the word's true prefixes and suffixes are removed to produce the stem or the root word [40]. This is important to classify texts and index. It is also used to make operations fewer dependents on particular form of words rather than enclosing all possible forms. The computational process gathers all words that share the same stem and have some semantic relations. It is also used for document matching and classification to convert all likely forms of a word in the input document to the form in reference document [40].

The types of automatic stemming strategies are four: affix removal, table look up, successor varity, and N-gram [36]. Affix removal stemming is imitative, simple and can be implemented efficiently.

In this type of strategy suffixes and/or prefixes will be removed from the word to get the stem. This type of strategy was used by [39, 36, and 40]. Table look up strategy looks for the stem of a word in a table of the dictionary. This strategy was used by [6] to stem Amharic morphological words. It is a simple procedure and depends on the size of stemmed dictionary before the activities will be performed. It also requires considerable storage space. Successor varity stemming is done based on the determination of morpheme boundaries, uses knowledge from structural linguistics, and is more complex than affix removal. N-gram stemming is used based

on the identification of bi-grams and tri-grams and is more of a term used for clustering than stemming. It is done based on numbers of n-grams. This strategy was used by [41] to stem language independent terms using uni-gram.

Unlike languages like English that has less morphological variants, Semitic languages like Amharic, Arabic, Ge'ez are much rich in morphology. So, like Amharic [38], Arabic [40], Ge'ez [44], Silt'e (which is one of the Semitic languages in Ethiopia) involves dealing with prefixes, infixes and derivatives in addition to suffixes.

1.2 Statement of the problem

According to [46], the EthioSemetic and South-Arabia languages are categorized under south Semitic. The EthioSemetic languages are further classified as North- EthioSemetic and South-EthioSemetic. Tigrigna together with Ge'ez and Tigre form the North- EthioSemetic language and the South- EthioSemetic includes Amharic, Argoba, Hariri, Silt'e, etc.

Silt'e is one of the zonal languages in the southern Nations, Nationalities and peoples region of Ethiopia. According to the 1998 census, the language is spoken in Central Ethiopia by about more than 827,764 people. Besides the Silt'e Zone, Silt'e is also spoken in Ethiopia's major cities, particularly in Addis Ababa. Elementary schools in Silt'e zone give all subjects in Silt'e from grade one up to four. After grade four, it is given as one subject for grades up to ten and also given for grade ten national student exam in Silt'e zone. The language uses Saba Ge'ez (Ethiopic script) [35]. Based on the researcher's preliminary analysis there are a significant amount of electronic documents in Silt'e in schools, government organizations of the zone and on net and there is an increase from time to time. However, there is no tool or system that supports the organizations to access valuable information for the purpose of decision making and other problem solving. Most of the time organizations are requested for information by other organizations and the request may need immediate response. The absence of tools or systems that help them in accessing and searching information delays their reply; as a result, they may fail to achieve their organizational goals.

Retrieval systems enable us to get information easily and timely. In today's shrinking world, it is becoming evident that there is a large body of information and research available only in the

language of the primary researcher, and much information can not be shared between research communities with out considerable translation and time delay [47].

Including Silt'e language, there are more than 86 languages in Ethiopia. But there are few research works done to design stemmer or an information retrieval system for local languages, such as Amharic [42], Tigrigna [40], Afanoromo [43], Wolayta [13], and Ge'ez [44] to simplify the problem of morphological complexity of the respective languages. Since most of the stemmers developed for other languages are language specific, it is not possible to make use of a stemmer developed for other languages. This is because of structural difference of the languages. So it seems quite difficult to follow the same stemming pattern and apply the same technique for all languages. Therefore, the stemmer of one language can not be effectively applied to other languages [23].

In developing an IR system for a particular language, the study of automatic indexing technique is important for the language. Among the process of automatic indexing, stemming and stop words are highly dependent on the document language. The motivation for using stemming is the need to increase effectiveness of retrieval system since stem of a term represents a broader notion than the original term itself [17]. To my knowledge so far no IR computational tools like stemming have been developed for Silt'e language. Therefore, it is felt necessary to undertake a research on the development of stemming algorithm for the language. Hence, this study.

It is, therefore, the aim of this study to explore the possibility of designing stemmer algorithm for Silt'e language. So this research attempts to answer the following questions:-

- What are the properties and word formations in Silt'e language?
- Is it possible to adopt some techniques from stemming algorithms developed for other languages?
- Does the designed prototype system registers good performance based on the sample text documents?

1.3 OBJECTIVES

The present research has the following general and specific objectives.

1.3.1GENERAL OBJECTIVE

The main objective of this study is to design a stemmer for Silt'e language.

1.3.2 Specific objectives

To achieve the above general objective the following specific objectives will be attempted in the research work.

- To have a clear understanding of the area through conducting relevant literature review and identify different stemming algorithms that have been developed for other languages.
- To review properties of the Silt'e language in order to get familiar with the different aspects of the language and knows how the separation of words is achieved.
- To build a stop word list and compile a list of affixes used for the corpuse.
- To designe appropriate stemming algorithm for Silt'e language.
- To develop a stemmer that identifies inflectional and derivational affixes of Silt'e language.
- To test the performance of the stemmer on the selected Silt'e text and report the finding of the study.
- Forward recommendations for further study.

1.4 METHODOLOGY AND METHODS

1.4.1 LITRATURE REVIEW

Document analysis used to understand the characteristics of the language. As studying the language's morphology constitute an important component in the research, a literature survey was made to gather information and to understand the language. Discussion with appropriate individuals (linguists) was not possible because of different resons.

1.4.2 DATA SOURCE

A text corpus is one of the resources required in stemming algorithm or IR researches. The text was used for compiling stop words, prefixes and suffixes. Moreover, the text corpus has been used for testing the algorithm. A good sized text can show a reasonable language morphological behavior. Selection of text is, therefore, an important component in developing a stemmer. For the purpose of this research texts from school books and other sources were used.

1.4.3 DESIGNING AND DEVELOPING TOOLS AND TECHNIQUES.

Python (3.1) programming language was used to develop the stemmer algorithm for Silt'e. This is because it is relatively easier to manipulate text (string manipulation) and also the researcher has some experience in programs using python programming language. Text documents (texts written in Silt'e, grammar research book of Silt'e language and school books) have been used to the study the language's characterstics. For the experiment context sensitive affix removal technique was used. This is because affix removal techenique is relatively easy and, efficient to implement than other techniques.

1.4.4 TESTING PROCEDURE

To evaluate the stemmer, manual error counting and vocabulary reduction method were used. Manual error counting method found to be appropriate and was used to test the stemmer. Dictionary reduction method also was used to test the compration power of the stemmer.

1.5 SCOPE OF THE STUDY

The main aim of this study is designing a stemmer for Silt'e language. This research mainly focused on derivational and inflectional word variants of the language. Compound and irregular words and infix stripping technique were not addressed here because of difficulty to handle and time limitation. The stemmer contains; prefix striping, suffix stripping and letter reduplication striper techniques (type 1 and type 2). Prefix-suffix striping is not included in the stemmer because it can be handeld by incorporating the prefix part to prefix list and the suffix to suffix list.

1.6 Significance of the study

One of the tools for IR is stemming. By developing a stemming algorithm for Silt'e variant words can be conflated. In Silt'e a word can have very large variants and conflating these variants increases retrieval performance. It also reduces storage of index files. In addition to the above results, it can also give the following benefits:-

- Can be used as one input to integrate and develop a general Semitic language information retrieval system and to form multi-lingual information retrieval.
- This research can also help to develop tools like spell checker, grammar checker, thesauri, word frequency counter, document summarizers and indexers. It can also be used to reduce word variants and to minimize total number of files.
- It can be used for developing cross lingual retrieval systems. For example one can feed a Silt'e text query to search engine, this entered word may be translated to other languages like English and access documents in English or other languages

1.7 ORGANIZATION OF THE THESIS

The research work consists of five chapters. This chapter introduces the importance of stemming on IR environment and the need to develop a stemming algorithm for conflating variants of a word in Silt'e language. Statement of the problem and its justification, methodology employed, and scope of the study are also discussed in this chapter.

Chapter two reviews the works on conflation techniques in general and stemming algorithms in particular. Detailed discussions are made on approaches to stemming and types of stemmers. Review is also made in this chapter on some stemming algorithms developed for other foreign and local languages.

Silt'e language morphology is reviewed in chapter three. The inflectional and derivational morphologies of the language are the main concerns of this chapter. Word formation processes for Silt'e nouns, adjectives, and verbs will also be presented in detail in the chapter.

Discussions on the development and, experimentation of the stemming algorithm for Silt'e text are given in the fourth chapter. The compilation of stopword lists and affix (prefix-suffix pair, prefix, and suffix) list is presented in this chapter. The approach employed to develop the stemmer and the reasons for its selection a lso parts of the discussions in this chapter.

The last chapter, chapter five, presents conclusions deduced from the findings and recommendations for future research.

CHAPTER TWO

REVIEW OF RELATED LITERATURE

2.1 CONFILATION TECHNIQUES

Queries and documents related with each other by the terms they have in common .The more they have similar terms the more they will be related. Therefore the degree of relationship between them is determined mainly by the frequency of terms they have in common [21]. However some words may have different forms that require some form of processing to change it to its root word. The existence of these variants for a word resulted from its different morphology. Hence morphological variance of a word is the main problem in indexing and retrieving systems [3].

Words can have different forms (variants) in a sentence for the sake of making agreement between subjects and verbs in gender, number, tense, person, mood or voice. Moreover, valid alternative spellings (e.g. “realize” and ”realise”), antonyms (“agreement” and “disagreement”) and different spelling use for the same word in different areas(often and color (America)and offen and coulor (Britain)). Besides abbreviations are also causes to word variants [22]. Generally, morphological variance is the main reason for text word variants [38]. Thus, morphology uses to analyze and describe the forms of words in a language, and usually includes inflectional and derivational.

Derivational morphemes result in new word that have different meaning from the previous one, like “creation” which is formed from “create” by adding suffix “–ion”. On the contrary inflectional morphemes don’t result in meaning difference between variants. An inflectional morpheme varies the form of words in order to express grammatical features, like singular/plural or past/present tense. For instance, student and students are two different forms of same word [42].

Compounding is the process of constructing morphologically complex words from two or more unbound morphemes. That is, compounding is the joining of two linguistic forms, which are functionally independent [43].

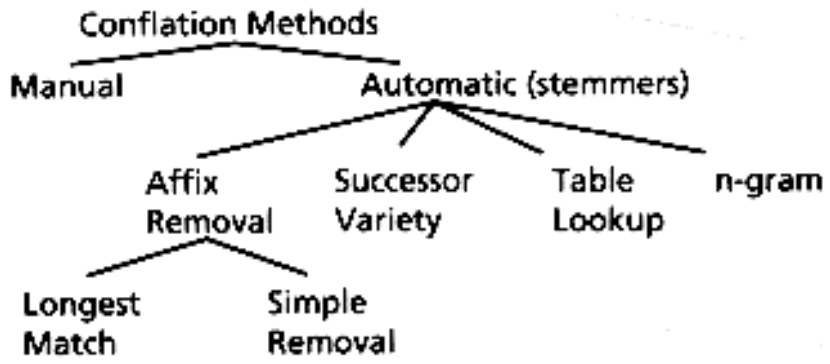
Morphological variants of a given word taken as analogous and, equivalent terms for the sake of information retrieval application. Hence, we have to take in to consideration the morphological variants of words as one of reference of all word forms. Converting these complex forms of words in to their root form is very important in determining the relevance of documents to user's queries in information retrieval. This concept helps to perform stemming in different languages using a number of stemming or conflation methods to get root/stem of morphological variants [21].

Different forms of a word are not recognized equivalently unless there is some kind of natural language processing (NLP).Because word form variations has significant impacts on information retrieval systems effectiveness [21]. Therefore, devising a mechanism to reduce a variation of word to its root form is important. To alleviate the word morphological problems, we use a method that helps to bring together semantically related words to a single form, which is a conflation technique [19].

Conflation is the process of identifying morphological variants of words that have similar concept and represent them by their root words [22]. It can also be defined as the process of matching morphological term variants by mapping term variants to a single form, usually a unique well-formed root for each word root for each word [24]. It performs by bringing similar words that have similar meaning with in common forms. Conflation techniques used to solve problems that are raised due to variations in word forms.

Conflation helps to increase efficiency and effectiveness of the information retrieval systems [40]. It also uses to understand the degree of relationship between different word forms, to overcome vocabulary mismatch problems in addition to better generalizations [20].

Conflation techniques discussed in [22, 19]: the two types of conflation techniques are manual and automatic. Manual conflation is performed at search time with right hand truncation. It is performed on query words but not on documents.



The other technique i.e. Automatic compilation is designed using stemmers. It alleviates the problem of word variants by matching different forms of the same words to their root automatically. Automatic conflation performed both on query and document words. This technique has two classes [23, 19]: stemming algorithm and string similarity algorithms. Stemming algorithms or stemmers are techniques for reducing words to their grammatical root stem form. These types of algorithms are usually designed to handle morphological variants with in language. In contrast, string similarity algorithm is usually language independent and designed to handle all types of variants. This approach involves conflating by calculating and measuring the similarity between an input query and terms and each of the distinct terms in the data base. Terms that have a high similarity to a query terms are then displayed to the user for possible inclusion in the query. The common example to string similarity algorithm is N-gram matching techniques.

In both manual and automatic conflations, over stemming and under stemming problems may occur. In over stemming, the stem is too short (too much of affix is removed from a term) and will result in unrelated words being stemmed to the same stem; while under stemming arises if too short affixes are removed and may result in related terms being described by different stems. Example; the stem of a word ‘instructors’ may be considered as ‘instructor’, rather than ‘instruct’ . These errors are occurred when a word is not stemmed properly. These causes irrelevant documents are retrieved or only few relevant documents are retrieved. The errors that may occur in stemming could be corrected manually using different exceptional lists, or by means of rules in the stemmer [20].

The effectiveness of the retrieval system is usually measured/evaluated using a combination of both recall and precision. Precision is a proportion of the retrieved documents that is usually relevant to the user interests; on the other hand, recall is a ratio of the relevant documents that is actually retrieved from the file. Achieving high recall and retaining a high precision use to improve the salient of information retrieval systems [33]. Hence, to improve the recall level, there are various types of methods such as use of term phrases, use of truncated terms or stems, and use of the synonyms [24].

Another issue that helps to illustrate retrieval effectiveness is stop word lists. Stop words are removed because of the following impacts on information retrieval [20]: they can affect the retrieval effectiveness and weighting process. This is because they mostly have a very high frequency and tend to diminish the impact of frequency differences among less common words. They can also affect efficiency due to their nature and the fact that they carry little meaning, which may result in a large amount of unproductive processing. Therefore, removing stop words results in increase effectiveness of retrieval by reducing storage requirements and increasing the matching of a query with index terms of a document.

2.2 STEMMING ALGORITHMS

Words in the query are matched with the words in the text database to retrieve relevant documents [1].A query can be Boolean, where Boolean operators used to connect words in the query and documents must satisfy the given condition to be retrieved; or ranked, where the documents with the highest statistical similarity to the query are retrieved as answers. There must be a mechanism for deciding whether a given query term corresponds to a given word in a document for both types of queries. The simplest is to allow exact matching only; for example, “bicycles" would match itself only and a document that contained “bicycles" but “Bicycles" would not be recognized as a match. Another simple correspondence method is case-insensitive exact match.

Stemming is a more general form of correspondence, in which query terms and document terms are reduced to a common root word by removing affixes. Terms match if they have a common root. For example, “trains" might be stemmed to “train", which would match “train", “training", and of course “trains". Each word thus has a set of conflations, that is, words that have the same

root. Several stemming algorithms have been proposed [2, 3, 4, 5, 6, 7, 8], based on different principles; each produces rather different sets of conflations.

Stemming has usually been measured (evaluated) by its impact on querying: since stemming changes the documents that are retrieved in response to a query, it has the potential to change the quality of the set of answers. Despite the fact that stemming and choice of stemmer can significantly change the number of documents that match a query term, the results of information retrieval experiments have been mixed, ranging from negligible difference in average performance to unambiguous but small improvements [9, 10, and 3]. However, measuring average performance can mask significant changes in individual queries, and the metrics used for assessing performance can allow the effects of stemming to seem small, even if the documents sets retrieved are changed significantly.

"Instead of using variation in precision and recall as an indirect measure of stemming accuracy, it is possible to evaluate stemming algorithms by the accuracy of the conflations they produce. This was the approach used by Paice [11].

Although the character of stemming algorithms may vary significantly depending on whether a stem dictionary is being used, whether a suffix list is being used, and of course on the purpose for which the stemmer is designed, most of them are based on certain principles and procedures [5,12]. These procedures involve either the removal of the single longest matching suffix or the iterative removal of several suffixes. The rational behind iterative approach is the fact that suffixes are attached to stems one after the other. In the iterative approach, suffixes are removed from the stem in the order of their derivational rules. In this approach the stripping starts from the end of the word and working towards the beginning. For instance, if the word FRUTEFULLNESS is considered, the suffix –NESS will be removed in the first iteration and re-considering FRUTEFULL, the suffix –FULL will be removed leaving FRUTE as a final stem. Stripping is done based on the class order that is defined by the programmer. Other approaches such as proposed by [11] may also exist, which are iterative but do not have their endings classified. Using the iterative approach has also its own problem [13] the need of examining large number of endings, which is a difficult activity and compilation of list of ordered class.

Iterative approach stripping procedure is used by Porter**.** It operates in five stages, using five different classes of suffixes (a total of about 60) to simulate the inflectional and derivational process of words. Some of the rule does not actually delete the suffix but they transform endings of the stem to new endings. For instance, if we remove the suffixes "**–ing"** from "absorbing" and "**- ion"** from "**absorption**", it gives us "**absorb**" and "**absorpt"** which are different stems. But using the recoding rule that transforms endings from "**–rpt"** to "**–rb**," the two words are conflated to the same stem that is "**absorb" [6]**.

In the case of longest mach approach, it removes the longest suffix possible. If the same word FRUTEFULLNESS is considered the suffixes in the word are: -NESS, -FUL, and FULNESS therefore the algorithm removes –FULNESS from the word. The problem of using longest match approach compared to iterative method is: need for generating all possible combinations of affixes and processing and storage space required and the change of affix during concatenation.

There are context-free or context-sensitive stemming algorithms. Context refers to any property that is attached to the remaining stem and usage of the suffix [14]. In context-free algorithms, no restriction is applied on the stem and hence no additional operations are required to check restrictions. But most of stemming works [5, 6,15, 16] indicated that better result could be obtained by adding constraints to stripping operations, that is, using context-sensitive, which are primarily language dependent. Context sensitive rules specify particular conditions in which each suffix may be stripped from an input word. Savoy [15] illustrates three general types of constraints: quantitative, qualitative and recording rule.

"In quantitative constraints minimum length for the remaining stem is set when a suffix is removed. This helps not to remove ending from a stem in which the ending is in the suffix list but actually not a suffix for that stem. For example for the word ABILITY and suffix ABILITY, as the remaining stem must not be zero, the suffix –ABILITY will not be stripped from the word."

Qualitative constraints define conditions to be satisfied by ending of the remaining stem. For example in English, a stem does not end with that is, stripping "ize" from "seize" is not allowed.

Spelling corrections and adjustment rules must be used to conflate the words to exact stem. A recording rule is a transformation of the form AxC →AyC, where A and C specify the context transformation, X is input word (string), and y is the transformation string.

There are stemming ranging from a weak stemmer, which eliminate only plural markers to a intricate one that removes suffixes and prefixes. From the available literature, stemming algorithms for local languages and abroad discussed well.

Most stemming algorithms follow the principles and approaches employed by the two most common English stemmers, Lovis and Porter. These two algorithms that form the bases of most algorithms employ suffix stripping.

Lovins [5] algorithm is based on longest match principle, and uses a list of 260 endings sorted in decreasing order of length. For instance, a word COMPUTATIONALITY would be stemmed to COMPUT if the suffix –ATIONALITY were included in the list. After a suffix is removed, the stem is compared with one of the rules (34 recoding rules) to consider spelling exceptions. The rules define, for instance, the treatment of a suffix preceded by double consonants (such as JUMPPING to JUMP), or minimal stem size must be retained (such as the removal of ING from PLAYING but not from SING) etc. the problem of using longest match approach is the need to have a largest affix list: basic and possible combinations. This requires storage costs and compiling and processing the list.

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 Techniques, successor variety, Affix removal and Statistical (N-gram).

2.2.1 DICTIONARY-BASED TECHNIQUES

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 affixation. Each word uses a unique entry in a look up table. For example terms such as engineering process is performed manually, wherein the stem stems are defined for each word and stored in some kind of structured form [60].

This techniques list all words that exist in a specific language which 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 association root form is returned. Terms from queries and indexes could then be stemmed via table look up. Table entries are listed in 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 being stored. The space occupied for storage tends to grow as the corpus expands, which can make the search process inefficient. In Korvetz's dictionary experiments [74], this direct method, seems effective but in adequate 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 [71] 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 [77]. 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 [77], iterative longest match stemmers have been reported by Dawson [1974], Porter [81], and Paice & Husk [79].

Affix removal algorithms are often called Rule-Based Technique and is a widely applied stemming technique and the algorithm is introduced by Porter [81]. 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 [77] 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 [78] and Hidden Markov Models (HMM) [84]. Most stemmers are language-specific. Single N-gram stemming suggested by [78] 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 [85] 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). Thai the sequence of characters before this point can as a stem. Using Porter's algorithm as a baseline, they found that HMM had a tendency over stem 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 [83]. 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 [70]. 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.

2.3.2 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 [57] which investigated the effect of stemming for information retrieval [86] 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 [80]. 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, recoding and context sensitive rules, with prefixes being removed before suffixes. Once the stem of the word is obtained, the root is obtained by stripping all the remaining vowels.

Those studies also indicated a positive effect from using the stemmed forms in information retrieval: Alemayehu and Willett 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 [57].

The other Amharic stemmer was developed by Atelach Alemu and Lars Asker [58]. 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 [13]. 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 [59] 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 [82], Slovene Language stemmer by [18], Arabic

Language stemmer by [73] 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 [77], [58], [81], [12], [79] and [60].

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 [77] 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 receding 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 matches the stems of like meaning words. The Stemmer does not satisfy from the IR viewpoint either, as its large rule set, and its receding 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 receding transformations.

2.4.1.2 DAWSON STEMMING ALGORITHM

Dawson stemming algorithm [58] is based on the one developed by Lovins [77] 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 [81]. 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 [79]. 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 [74]. 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 receding, 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 [76]. 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 [73]. 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 [73].

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 [72] , and Larkey et al.'s U Mass Stemmer [75].

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 stop word 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 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 [87]. Accruing to his method, measures of under stemming and over stemming 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 under stemming 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 over stemming errors, which reduce precision. A good stemmer should therefore produce as few under and over stemming errors as possible.

CHAPTER THREE

MORPHOLOGY OF SILT'E LANGUAGE

3.1 INTRODUCTION

Stemming is a fundamental step in processing textual data preceding the tasks of information retrieval, text mining, and natural language processing. The common goal of stemming is to standardize words by reducing a word to its base (root) [51].

Like the other Semitic languages (Amharic, Tigrigna, Ge'ez etc) Silt'e language has the feature of transmitting different messages with a single word alone. Since a single word can have different forms by adding affixes or changing alphabetical orders (patterns), this situation makes the language morphologically rich[52].

The main purpose of stemmer is automatically conflating different words that have similar meaning to their root by removing their affixes and/or rearranging the vowel patterns. Morphology plays an important role to guide the development of the stemmer, and describe the nature and characteristics of affixes in Silt'e words. Therefore developing (designing) a stemmer for a language requires analyzing, understanding and also modeling the language trend in terms of word formation. The above concept shows the importance of studying the Silt'e morphology for the purpose of modeling it and developing automatic procedures for the language. Thus, this chapter deals with the concepts of inflectional and derivational morphologies of Silt'e language.

3.2 THE WRITING SYSTEM OF SILT'E LANGUAGE

Since the 1980s, Silt'e has been written in the Ge'ez, or Ethiopic, writing system, originally developed for the now-extinct Ge'ez language and most known today in its use for Amharic and Tigrigna [52]. However, Silt'e vowels differ considerably from the typical set of seven vowels in languages such as Amharic, Tigrigna (see Table 3.2).

Ethiopic Fidel has the phonetic structure (order) of seven columns (see Table3.1).

See lists of constant, numbers, punctuations and vowels of Ethiopic (Ge'ez language) in appendix 1

First order (ግእዝ)	Second order (ካዕብ)	Third order (ሳልስ)	Forth order (ራብዕ)	Fifth order (ሃምስ)	Sixth order (ሳድስ)	Seventh order (ሳብእ)
(አ \ä\	ኡ \u\	ኢ \i\,	ኣ \a\,	ኤ \e\	እ \ə\	ኦ \o\
(ሀ \hä\	ሁ\hu\	ሂ \hi\,	ሃ \ha\,	ሄ \he\	ህ \hə\	ሆ\ho\

Table 3.1: Table shows the seven alphabetical order of Ethiopic (Ge'ez) alphabet.

3.2.1 VOWELS AND CONSONANTS OF SILT'E LANGUAGE

The Ge'ez script makes distinctions among only seven vowels, so some of the short-long distinctions in Silt'e are not marked. In practice this probably does not interfere with comprehension because there are relatively few minimal pairs based on vowel length. In written Silt'e, the seven Ethiopic vowels are mapped onto the ten Silt'e vowels as follows: the five short vowels are አ \a\, ኦ \o\, እ \i\, ኡ\u\, ኤ\e\ and the five long vowels are ኣ \aa\, ኦኦ\oo\, ኢ\ii\, ኡኡ\uu\, ኤኤ\ee\. Only the short and long **a** and the short and long i have been distinguished in the alphabet. The short **a** is indicated by the first order (form) of the Ethiopic script and the long a by forth order (form). The short i and consonant alone are indicated by sixth order (form) and the long i is indicated by the third form. The third form is also used word finally when the word ends in i [52]. Table 3.2: illustrated more the vowel mapping by using practical words that show the vowel mapping in Silt'e language more clearly with simple example of order of alphabet.

The order (form) of Silt'e alphabets 1 2 3 4 5 6 7
ሀ \ha\ ሁ\hu (huu)\ ሂ\hii\ ሃ\haa\ ሄ\ he(hee)\ ህ\hi\ ሆ\ho (hoo)\
ነ \na\ ኑ\nu(nuu)\ ኒ\nii\ ና\naa\ ኔ\ne(nee)\ ን\ni\ ኖ\no(noo)\
ኘ\ña\ ኙ\ñu(ñuu)\ ኚ\ñii\ ኛ\ñaa\ ኜ\ñe(nee)\ ኝ \ñi\ ኞ\ño(ñoo)\

- ***ä* → *a***: **አለፈ** |*alafa* | 'he passed'
- ***u* → *u*, *uu***: **ሙት** |*mut* | 'death', |*muut*| 'thing'
- ***i* → i**
  - *ii*: **ኢን** |*iin*| 'eye'
  - word-final *i*: **መሪ** |*mari* | 'friend'
  - *i* ending a noun stem: **መሪከ** |*marika*| 'his friend'
  - impersonal perfect verb *i* suffix: **ባሊ** |*baali* | 'people said'; **በባሊም** |*babaalim*| 'even if people said'
- ***a* → *aa***: **ጋራሽ** |*gaaraaš*| 'your (f.) house'
- ***e* → *e*, *ee***: **ኤፈ** |*eeffe*| 'he covered'
- ***i* → ə**
  - *i* (except as above): **እንግር** |*ingir*| 'foot'
  - consonant not followed by a vowel: **አስሮሽት** |*asroošt*| 'twelve'
- ***o* → *o*, *oo***: **ቆጨ** |*k'oč'e* | 'tortoise', |*k'ooč'e* | 'he cut'

Table3.2: The mapping of seven Ethiopic vowels to ten of Silt'e and order of Silt'e alphabet

Silt'e has a fairly typical set of consonants (26 consonants including **ፐ** |**p**|) for an Ethiopian Semitic language. There are the usual ejective consonants, alongside plain voiceless and voiced consonants and all of the consonants, except /h/ and /ʔ/, can be geminated, that is, lengthened. The symbols /p/ and /ʔ/ (glottal stop) play only a marginal role in the system, |**p**|, because it appears in only a few words and |**ʔ**|, because (as in Amharic), it is often omitted. The consonants of Silt'e shown in Table 3.3:

ሀ |**h**| ,**ለ**|**l**|, **መ**|**m**|, **ረ**|**r**|, **ሰ** |**s**|, **ሸ**|**š**|,**ቀ**|**ḳ**|, **በ**|**b**|, **ተ**|**t**|, **ቸ**|**č**|, **ነ**|**n**|, **ኘ**|**ñ**|, **አ**|**a**|, **ከ**|**k**|, **ወ**|**w**|, **ዘ**|**z**|, **ዠ**|**ž**|, **የ**|**y**|, **ደ**|**d**|, **ጀ** |**j**| ,**ገ** |**g**|, **ጠ** |**ṭ**|, **ጨ** |*č'*|, **ጰ** |**P**|, **ፈ** |**f**| , **ፐ**|**p**|

Table 3.3: consonants of silt'e writing system

3.3 MORPHOLOGY

Morphology is a branch of linguistic that studies and describes how words are formed in language[55]. Similarly, [**58**] defines morphology as the study of the structure of words. There are two kinds of morphology: inflectional and derivational. Inflectional morphology is concerned

with the inflectional changes in words where word stems are combined with grammatical markers for things like person, gender, number, tense, case and mood. Inflectional changes do not result in changes of parts of speech. Derivational morphology deals with those changes that result in changing classes of words (changes in the part of speech). For instance, noun or an adjective may be derived from a verb.

3.3.1 SILT'E MORPHEME

A morpheme is a minimal linguistic unit of a language that carries meaning and that can not be further decomposed in to meaningful units [54]. A morpheme in Silt'e can be free or bound, where a free morpheme can stand as a word on its own where as a bound morpheme can not occur on its own as a word, for example these are free morphemes of Silt'e:- ሰብ|sab| 'person', ጋር|gaar| 'house', ቃዋ| ቈaawaa| 'coffee', they can stand by themselves. In contrast to these bound morphemes of silte can't stand by them selves. For example we can add suffix ች፡ |ca| to the word ሰብ |sab| to show paucal number; therefore ች፡ is bound morpheme of Silt'e [56].

3.3.2 WORD FORMATION IN SILT'E

Affixing (adding suffix, prefixes and infixes), compounding, duplicating (reduplicating) and changing vowel patterns are used to give different word forms in Silt'e language [53].

The addition of suffix, prefix and infix are the usual ways of word formation in Silt'e. Although it is not common, the addition of prefix and suffix at the same time is also used to form word variant. The possibility of prefixing and suffixing more than three prefix or/and suffix result in long word formation. Hence, the complex morphological structure of a single Silt'e word can give very large number of variants.

The application of compounding is another means to contribute in the process of word variant formation in Silt'e language. A compound is made up of two or more simple words (roots) and has a meaning that is different from that of its parts: አባዴ|abbaadde/abaade|, 'parents' from አቦ|abbo/abo|'father' and አዴ|adde/ade| 'mother'. ወዘነቁብሌ|wazana-ቈuble|'faint-hearted', ወዘነ|wazana| 'heart' and ቁብሌ| ቈubla|'deficent'.

Duplication is also the other means which is used to form word variation: there are at least two distinct kinds of reduplicative formations in Silt'e language: type one involves the reduplication of the first letter (that is ላ |laa|) ላላሀ|laalaaha| 'send many times' formed from ላሀ |laaha| 'send', type two by reduplicating the second letter by using its forth form (that is መ|ma|): ጅማመረ |jimaamara| 'starts' to little which is formed fromጀመረ |jamara| 'start'.

3.4 INFLECTIONAL AND DERIVATIONAL MORPHOLOGY OF SILT'E

Inflectional affixes describe word as stems are combined with grammatical markers for things like person, gender, number, tense, and case. There are five kinds of speech in Silt'e: adjectives, nouns, verbs, adverbs, and propositions [53]. Prepositions and conjunctions are totally unproductive for IR purpose or other natural language processing. The same thing happens for adverbs and is few in number. Therefore, the discussion of derivational and inflectional morphology concentrates on the remaining three parts of speech, namely verbs, nouns and adjectives.

To study the morphological nature of the language, the researcher used different sources such as [53], [52], [57] and [56]. The translation of the Latin words of the available document that talks about Silt'e morphology to Ethiopic-script is performed and prepared by the researcher. The following morphological analysis part adopted (taken) from [53].

3.4.1 NOUN'S INFLECTION AND DERIVATION

3.4.1.1 NOUN'S INFLECTION

Nouns in Silt'e language inflected for definiteness, gender, number, and case. Morphologically nouns have the following general structure in Silt'e language:

<prep><distrib>* stem *<acc> <part> <poss> <art> <voc> <cop>

3.4.1.1.1 DEFINITENESS, NUMBER AND GENDER

Definiteness, number and gender are interrelated in Silt'e language. Silt'e has two definite articles, the suffixes -ኢ|–ii| (-ይ|-y| after vowel) for masculine, and the -ቴ|–te| for feminine. However, these two articles can fulfill further functions; the masculine article -ኢ|–ii|can signal that one is dealing with a large number or collective of items, regardless of their gender, and the

feminine article can indicate that one is dealing with one item (singularize) and /or with a small specimen. For example, in addition to denoting definiteness, the expression **ጋርቱ|gaarte|** 'the (f) house' indicates that one is talking about a single house, and also that the house is considered to be small. By contrast , the expression **ጋርኡ|gaarii|** 'the (m) house ' with the masculine definite marker is neutral with regard to size, and it can refer to a collective or large number of houses rather than one.

Gender is mostly determined naturally, not grammatically. Thus humans and animals are given the gender according to their sex. A comparatively small number or words referring to inanimate objects are specified for gender; thus **ወር|*wari*|** 'moon' is usually given masculine gender and **አየር|*ayr*|** 'sun' feminine, hence **ወሪ|*warii*|**'the moon' **አየሪቱ|*ayrite*|** 'the sun'. Similarly trees are usually treated as **feminine**. The collective use overrides natural gender: **እንዳቺ መጠ |*Indaaccii mata*|** 'the woman have come'. The gender markers for possessive pronouns shown in table 3.4:

1st p		2nd p				3rd p			
singular	plural	Singular		plural		singular		Plural	
		m	F	M	f	m	f	m	f
ኤ|ee|	ና|na|	ኣ/ኣሀ |aa/aaha|	ኣሽ |aas|	አሙ |ammu|		ክ|a|	ሽ |sa|	ኒሙ|niimmu|	

Table 3.4: Possessive suffix for noun inflection

Table 3.5 shows samples ilustrate combinations of several of these suffixes:

አደኔውከ
Aaddeenawka ‘as to his mother <acc.>’
አደ ኔ ው ከ
Aadde -na -w -ka
Mother acc. Pragm. Marker -w his
ልክኔ
Likkinee ‘it is my measure/size’
ልክ ን ኤ
Likk -n -ee
Measure copula 1st sg-poss

Table 3.5: example of combination of suffixes

Semantically Silt’e has three forms for number singular, paucal, and plural. However, morphologically only the paucal is marked – the other two categories are unmarked. Thus while **ኡንች**|uunca| refers to a small number of stones <paucal>, **ኡን**|uun| refer either to one or large number of stones. Context is used to disambiguate such unmarked expression. Table 3.6 shows Noun paucal formation with suffixation of -**ች**|ča|:

Noun	Gloss	Suffixes	Paucal
ቻፍ|Caaf|	‘leaf’	-**ች** |ča|	**ቻፍች**|caaf ča|
ሰብ|Sab|	‘person’	-**ች** |ča|	**ሰብች**|sabča|
በርት|Bart|	‘stick, club’	- **ች**|ča|	**በርት**ča|bartičča|
ቦላሌ|bolaale|	‘long trousers’	- **ች**|ča|	**ቦላልች**|bolaalča|
ጦላቢ|toollaabi|	‘beggar’	- **ች**|ča|	**ጦላብች**|Toollaabča|

Table3.6: Noun paucal formation with suffixation of -**ች**|ča|

There are two basic morphological types of paucal formation for nouns; suffixation of -ች፡|– **č**a| **<-cča** after consonant cluster of geminates> and reduplication of the last consonant.

As you can see from table 2, word final **short vowels** are omitted before the paucal suffix.

Many nouns – mostly ending in a short vowel – form the paucal by reduplicating the last consonant, inserting the vowel ኣ|aa| and replacing the original final vowel by ኦ|o|. When the last consonant is marked, only the degeminated, single consonant is repeated.

Most words that form the paucal by reduplication allow the alternative of reduplication plus suffixation as well. Table 3.7 shows paucal noun formation with reduplication and reduplication plus suffixation:

Noun	Gloss	Paucal with reduplication	Paucal Reduplication+suffix
ቦሶ|boso|	'Young enset plant'	**ቦሳሶ**|bosaaso|	**ቦሳስች**|bosaas **č**a|
አለገ|alaga|	'Stranger'	**አለጋጎ**|alagaago|	**አለጋገች**|alagaag **č**a|
አሞሌ|amole|	'salt bar'	**አሞላሎ**|amoolaalo|	**አሞላለች**|amolaal**č**a|
ቡሬ|bure|	'big tin, big can'	**ቡራሮ**|burraaro|	**ቡራረች**|buraar**č**a|
በለ|bala|	'Calamity'	**በላሎ**|balaalo|	**በላልች**|balaal**č**a |

Table 3.7: paucal noun formation with reduplication and reduplication plus suffixation

3.4.1.1.2 CASE

Syntactic case relations are marked either by prefixes (some of which may also be considered prepositions – see below) or suffixes: the accusative case is marked by ኘ|na|, ኣ|a| (ኣ|-a| after consonants); the genitive by የ|ya-|; the dative/benefactive relation by ለ|la-|; the instrumental or determinative relation by በ|ba-|, the ablative by ተ|ta-|, and the vocative by ኦ **|-o|.**

3.4.1.1.3 MORPHOLOGICAL PROPERTIES AND PROCESSES WITH NOUNS

Nouns can end in any consonant or semivowels <except the glottal stop>, single or geminate, subject to the general phonological constraints of the language, or in one of the short vowels አ| a|, ኤ|e|, እ|i|, or ኦ|o| (no noun ending in ኡ |u| has been found.) The following morphophonemic changes are common with prefixes: Cአ+ወ→Cኦ/Ca+wa→Coo (**here C is to represent constant letter**) – የ|ya| plus ወሮ|waro| 'dragon' : የ·ሮ|yooro |'of a dragon'; Cአ+እ→Cኢ /Ca+i →Cii – ለ|la-| plus እንባብ|inbaab| 'snake': ሊንባብ |liinbaab| ' for a snake': (**here V represent vowel letter**) Cአ+V→ CV/ Ca+V → CV: በ|ba-| plus አደን|adan| 'cat' ባደን|baadan| 'by the cat '; ለ|la-| plus ኡ·ም|umi| 'uncle <maternal>' ሉ·ም|luumi|. Common morphophonemic changes with suffixes are the following. Word final vowels are dropped directly before paucal suffix (illustrated above in table2 and 4). Word final vowels are lengthened immediately preceding the acc. suffix -ን|–na|: ማጠ|maata|, ማጣን|maataana|; preceding the pragmatic marker -መ|–m| and ወ|–w|: አዴኘ|addenna|; አዴኛም|addennaam|; preceding the masculine definite article: ቡ·ቾ| buco|, ቡ·ቾይ|bucooy|.

3.4.1.2 NOUN'S DERIVATION

Nouns can be derived from verbs. The infinitive has already been introduced above. Nouns of agent can be formed in two ways. Verbs ending in a palatal radical add the suffix ኢሎ |–iilo| to the perfective stem. Table 3.8 shows some example:

Note that the suffix may be added to the palatilized or non-palatalized form of the verb stem.

Non–palatal verbs add the suffix እ|–i| to the last root consonant of the perfective stem, palatalizing that consonant where possible; the last, non –thematic vowel is lengthened to ኣ| aa|: ቆመረ|qoommara| 'be strong'; ቆመሪ|qoommari| 'strong'; አመሰለ|amasala| 'pretend' መሳይ|amasaay| 'pretending'**.** Table 3.8 shows noun derivation from perfective stem by adding the suffix ኢሎ |–iilo|:

Table 3.8: noun derivation from perfective stem by adding the suffix **ኢሎ** |–iilo|

3.4.2 ADJECTIVE INFLECTION AND DERIVATION

Adjectives are inflected for number and derived from nouns. Actually most of the nouns in Silt'e can be use as an adjective.

አሞቄ|Amoqe| 'work reluctantly and badly'		**አሞቂሎ**|Amoqiilo|	'relectant worker'
ዛጨ|Zaace | 'watch look after'		**ዛጪሎ**|Zaaciilo|	'watch man'
በቼ|Bace| 'weep, cry'	**በቺሎ**|Baciilo| or **በኪሎ** |bakiilo|		'some one who craies easly'

3.4.2.1 ADJECTIVES'S INFLECTION

There are two basic morphological types of paucal formation for adjectives like nouns do. Suffixation of -**ች**\|– **č** a <-**cč**a after consonant cluster of geminates> and reduplication of the last consonant. Tables [3.9 and 3.10] show the two types of paucal formation for adjectives:

Adjective word	Gloss	Suffixes	Paucal
ቻፍ|Caaf|	'leaf'	-**ች** |**č**a|	**ቻፍች**|caaf **č**a|
በርት|Bart|	'stick, club'	- **ች**|**č**a|	**በርት**·**č**a|barti**čč**a|
ጦላቢ|toollaabi|	'beggar'	- **ች**|**č**a|	**ጦላብች**|Toollaab**č**a|

Table 3.9: Adjective paucal formation with suffixation of -**ች**\|**č**a\|

Adjective word	Gloss	Paucal with reduplication	Paucal Reduplication+suffix
አለገ|aalag|	'Stranger'	**አለጋጎ**|alagaago|	**አለጋገች**|alagaag **č**a|
ጉመራ|gumara|	'white'	**ጉመራሮ**|gumararo|	**ጉመራርች**|gumaraar**č**a|

ቡሬ|burre|	'big tin, big can'	ቡራሮ|burraaro|	ቡራረች|buraarča|
በለ|bala|	'Calamity'	በላሎ|balaalo|	በላልች|balaalča |

Table 3.10: paucal adjective formation with reduplication and reduplication plus suffixation

In Silt'e most of the nouns used as an adjective. There are no special adjectival forms for comparison; if the point of comparison is explicit it occurs in the ablative case with the suffix -ኮ\|–ko\|. However, very often comparison is expressed by verbs.

E.g. ኡሀ ተሂኮ መኒጃም /Uha tahiko manijaam 'He is prouder than I'

3.4.2.2 ADJECTIVES'S DERIVATION

Simple adjectives have no special structural characteristics: ፈየ\|fayya\|'good' ቦዝ\|booz\| 'bad' የሮሬ\|yaroore\| 'big' ቀል\|qall\| 'small'. **Adjectives can be derived from nouns in a number of different ways**. None of these formations seems to be generally productive, but they are all lexically determined. Table 3.11 shows the derivation of adjectives from nouns:

Denominal adjectives in -አ'ኘ |-anna|: **ጉ·ት**|Gutt| 'middle' **ጉ·ተ·ኘ**|gutt**anna**| 'average'
Denominal adjectives in –አተ'ኘ|-atanna|: **ከስብ**| Kasb| 'work' **ከስበ·ተ·ኘ** |kasb**atanna**| 'industrious' **ፈ·ሮ** |Faro| 'injestice' **ፈ·ሮ·ተ·ኘ** |faro**otanna**| 'unjust'
Denominal adjectives in -ኣንቺኦ|aancio|: **ብ·ተ·ር** |Bitar| 'marriage' **ብ·ተ·ራ·ን·ቾ** |bitar**aanco**| 'newly –wed'
Denominal adjectives in –አ|-a| (-አ|-al) **-ም|-m| (-አ|-a|):** **መ·ኒ·ጀ** |Maniija| 'pride' **መ·ኒ·ጃ·ም** |maniija**am**| 'proud'
Denominal adjectives in -አ|-a|(አ|-al)ታ·ም|(a) |taam| **አ·ይ·ብ** |Ayb| 'milk' **አ·ይ·ባ·ታ·ም**|ayba**ataam**| 'giving much milk'

Table 3.11: The derivation of adjectives from nouns

3.4.3 INFLECTIONAL AND DERIVATIONAL MORPHOLOGIES OF VERBS

Following the typical Semitic root-pattern morphology, the lexical meaning of most verbs is specified by the consonants of the root. Underived verbs have two to four root consonants, or radicals. (Since the glottal stop is rarely pronounced, it is not written here. However inflectionally these seemingly vowel -initial roots behave exactly as if they had an initial root consonant.) Table 3.12 shows examples of underived verbs with 2, 3 and 4 radicals.

II – RADICAL VERBS:	**III-radical verbs:**	**IV-radical verbs:**
ገ·በ |Gaba| 'enter'	**ሀ·ረ·ተ**|harata| 'swallow'	**ድ·ነ·በ·ተ** |Dinabata| 'be surprised'
በ·ደ|bada| 'take'	**ረ·ወ·ተ**|rawata| 'run'	**ስ·ነ·ተ·ለ** |sinatala| 'know'

Table 3.12: examples of verbs from 2 to 4 radicals

The final radical of a verb can be of four kinds: ending in a consonant only, ending in consonant followed by **አ**\| a\|, ending in a non-palatal consonant with following **አ** \|a\| palatalized to **ኤ**\|e\|.

The nature of the final radical can best be seen in the imperative masc. singular; palatal consonants are 'depalatalied' preceding back vowels, so the underlying non-palatal consonant appears, for example, preceding the infinitival ending –**ኡት**|-oot|. Table 3.13 shows the nature of final radical:

(a) CONSONANT ONLY:			
ዋበ |waaba |	**ዋብ** |waab|	'give!'	
(b) Consonant plus **አ** |a|:			
ቁረ |qeera|	**ቂረ** |qiira|	'watch'	
(c) Non-palatal consonant plus **ኤ**|e|:			
ኖዜ |noze|	**ኑዝ** |nuz|	'be anger'	
(d) Palatal consonant plus **ኤ**|e|:			
ሰቼ |sace |	**ስች** |sic|	**ስኮት**|sikoot |	'to drink'

Table 3.13: the nature of final radical of verbs

The nasal element appears in two and three radical verbs: **አንዜ**|anze| 'see', **ኤንዘ**|eenza| 'hold',**ኦንቴ** |oonte| 'close', **አንደረ** |andara| 'spent the night', **ኤንቀፈ**|eenqafa| 'embrace' ; most of these verbs begin with a vowel (underlying preceded by a glottal stop); however , at least one verb has been found that begins with a consonant: **ሶንቸ** |soonce| 'smell'. Verbs with a nasal element are special in that the nasal does not count as a radical consonant: thus **አንዜ**|anze |'see' inflects like a II- radical, and **አንደረ** |andara| 'spend the night' like a III-radical verb, though their roots have three and four consonants respectively.

3.4.3.1 ASPECT

Each verb has three stem forms: perfective stem, imperfective stem and a third stem not marked for aspect, hence referred to here as the non-aspectual stem. The perfective stem is used to form simple past, present perfect and past perfect tenses, the imperfective stem for the formation of the present/future and past-imperfective tenses; the non-aspectual stem serves for infinitive and imperative. Table 3.14 shows examples of the three forms of the verb:

Perfective stem	**መሰክ** |masak-|		
(a) Simple past	**መሰከ** |masaka|	‘he guided’	
(b) Present perfect	**መሰካን** |masakaan|	‘he has guided’	
(c) Pluperfect	**መሰከ ናር**| masaka naar|	‘he had guided’	
Imperfective stem	**መስክ**|mask-|		
(d) Present /future	**እ+መስካን** |i+maskaan|	‘he guides/will guide’	
(e) Continous past	**እ+መስክ ናር**| i+mask naar |	‘he was guiding’	
Non –aspectual stem	**ምሰክ**|m (i)sak-|		
(f)Imper. /jussive	**ምሰክ** |misak|	‘guide!’ (2 m sg)’	
	የምሰክ|Yamsak|	‘let him guide!’	
(g) Infinitive	**ምሰኮ** |misakoot|	‘to guide’	

Table 3.14: the perfective, imperfective and non-aspectual forms of verb

The stem formation makes use of two distinct morphological means: changes in consonant-vowel pattern of the root and rising of the thematic vowel. Thus Silt’e verbs can be divided in to two broad classes:

Class 1: All verbs with thematic **አ**|a|; taking the perfective stem as basic, the other two are derived by omitting the first and the second vowel respectively. Table 3.15 and table 3.16 show the pattern of consonant and vowels in class 1 verbs and examples of class 1 verbs of Silt’e:

Perfective	**imperfective**	**non – aspectual**
<C>CVC<VC>	**<C>CVC<C**	**CC<VC>**
		CCC<C>

(**C** represents one or more consonants of a word; **V** represents one or more of vowel sequence)

Table 3.15: the pattern of consonant and vowel in class one verbs

ሰቼ|Sac-e|	‘drink’	እሰቻን | i-sac-aan|	ስች | sic|
በተከ |Batakа|	‘pull out’	እበትካን| i-batk-aan|	ብተክ| bitak|

Table 3.16: examples for class 1 verbs of Silt’e

Case 2: class two verbs are all verbs with a thematic vowel other than አ\| a̱\|. Thematic ኤ \|e\| is raised to እ \|i\|, ኦ\|o\|, to ኡ\|u\| in the non- aspectual stem: Table 3.18 shows examples of class 2 verbs:

ጀጀ|Jeeje|	‘reaches’	እጀጃን| i-jeej- aan|	ጂጅ | jiij|
ኖዜ |Noze|	‘be angry,	እኖዛን| i-noze-aan |	ኑዝ |nuz|

Table 3.18: examples of class 2 verbs of Silt’e

Raising verbs with thematic ኣ \|aa\| are exceptional in that the ኣ\| aa\| is raised to ኢ \|ii \|in the imperfective aspect, see these examples:

ጫመ \|Caama\| ‘test well’ እጨማን \| i-ciim-aan\| ጫም \|caam\|

ላሀ\|Laaha\| ‘send’ እሊሃን \|i-liih-aan\| ላህ \| laah\|

(Note that not all verbs with thematic ኣ\|aa\| undergo rising; ራጀ\|raaje \| is an example: እራጃን\| i-raaj-aan\|.)

Class 2 verbs with more than two radicals also omit the vowel of the last radical in both the imperfective and non- aspectual stem forms, see these examples:

ሼበለ \|Seebala\| ‘dance’ እሼብላን\|i-seebl-aan \| ሺብል \|siibl\|

ቶቀሰ\|Tooqasa \| ‘beg’ እቶቅሳን\| i- tooks –aan\| ቱቅስ \|tuuks\|

3.4.3.1.1. VERB INFLECTION

The inflectional structure of Silt’e verbs is as follows:

<neg ><pers> STEM <pers> <ben/detr> <cps> <aux>

The double occurrence of the person marker indicates that Silt’e marks person by both preffix- and suffixes in forms based on the imperfective and non-aspectual stems.

Like in other Ethio-Semetic languages, sentence constituents other than the subject can optionally be marked in the morphology of the verb, as is indicated by the benefactive (ben), detrimental/instrumental (detr) and complement person suffix (cps) categories above. The complement person suffixes can occur without either the benefactive or detrimental/instrumental markers, but these last two categories require the presence of the person markers.

Table 3.19 shows the suffixes that used to form the present time are as follows:

1st p				2nd p				3rd p			
Singular		Plural		singular		plural		singular		plural	
m	F	m	F	m	f	m	f	m	f	m	f
አሀ፡- aahu|	አወ፡- aaw|	አን|- aan|		አሀ|- aaha|	አሽ|- aas|	አመ፡- aammu|		አን|- aan|	አት፡- aat|	አን|- aan|	

Table 3.19: suffixes used to form the verb in present tense

The Impersonal አን \|–aan\|, the past auxiliary is ናር \|naar\|- (past tense form of verb of existence).

3.4.3.1.2. PERFECTIVE

In Silt'e verb, the perfect nature of verbs uses to create the past and complete actions. These types of verbs are the bases to other forms that may inflect to their base. The suffixes in table 3.20 are added to the perfective stem to mark the subject in the verb:

1 sg	ኩ|-ku| after C(consonant)
	ሁ|-hu|, ዉ|w| elsewher
1p	ና|-na|
2f	ሽ|-s|, word finally
	ሺ|-si| elsewhere
2m	ክ|-ka| after C(consonants)
	ህ|-ha|, ኣ|aa |elsewhere
2p	ኩሙ|-kumu| after C
	ሙ|-mmu| elsewhere
3m	ኤ|-e| in palatal verbs
	ኣ|-a	elsewhere
3f	ጥ|-ta| preceding object suffix
	ት|-t| elsewhere
3p	ኡ|-u|
Impers.	እ|-i|

Table 3.20: suffixes added to the perfective stem to mark the subject in the verb

The category 'impersonal' in the last row of the table is used to indicate what some unspecified group of individuals or people in general do. Hence it is glossed here 'people'.

3.4.3.1.3. SIMPLE PAST

The suffixes in table 3.20 added to the perfective stem to form the simple past tense of the verb. Table 3.21 shows the simple past form of a verb:

Simple past

Simple past of C – verb	Suffix	Gloss
መሰኩ |masakku|	-**ኩ**|-ku|	‘I guided’
መሰከ |masakka|	-**ከ**|-ka|	‘You (2m) guided’
መሰክሽ |masaks|	-**ሽ**|-s	‘You (2f) guided’
መሰከ |masaka|	-**ከ**|-ka|	‘He guided’
መሰክት|masakt|	-**ት**|-t	‘She guided’
መሰክነ |masakna|	-**ነ**|-na|	‘We guided’
መሰኩሙ|masakkumu|	-**ኩሙ**|-kumu|	‘You (pl) guided’
መሰኩ|masaku|	-**ኩ**|-ku|	‘They guided’
መሰኪ|masaki|	**እ**|-i|	‘People guided’

Table 3.21 Simple past form of verbs

Simple past of palatal verbs	Gloss
ከሼሁ|Kaseehu/w|	‘I wanted’
ከሼሀ|Kaseeha|	‘You (m) wanted’
ከሼሽ|Kasees|	‘You (f) wanted’
ከሼ|kase|	‘He wanted’
ከሼት|Kaseet|	‘She wanted’
ከሼነ|Kaseena|	‘We wanted’
ከሼሙ|Kaseemmu|	‘You (p) wanted’
ከሱ|kasu|	‘They wanted’
ከሲ|kasi|	‘People wanted’

Table 3.22: Simple past form of palatal verbs

Simple past of Ca-verbs	Gloss
በላሁ/በላው|Balaahu/w|	'I ate'
በላህ|Balaaha|	'You (m)ate'
በላኽ|Balaas|	'You (f)ate'
በለ|Bala|	'He ate'
በላት|Balaat|	'She ate'
በላነ|Balaana|	'We ate'
በላሙ|Balaammu|	'You (pl)ate'
በሉ|Balu|	'They ate'
በሊ|Bali|s	'People ate'

Table 3.23: Simple past form of Ca-verbs (constant followed by vowel a)

3.4.3.1.**4. PRESENT PERFECT**

Two further tenses can be formed from the perfective stem: the present perfect and the pluperfect by adding the present and the past tense auxiliary respectively. Added to the simple past tense forms given above tables, they yield the set of present perfect forms in Table 3.24:

Present perfect form	Gloss
መሰከ·|masakkoo|	'I have guided'
መሰካ|masakkaa|	'You have (2m) guided'
መሰክሺ|masaksees|	'You have (2f) guided'
መሰካን|masakaan|	'He has guided'
መሰክታ·ት·|masaktaat|	'She has guided'
መሰክናን|masaknaan|	'We have guided'
መሰከ·ሞ·ሙ·|masakkumoommu|	'You have (pl) guided'
መሰኮን|masakoon|	'They have guided'
መሰኬን|masakeen|	'People guided'

Table 3.24: present perfect form of verbs

These forms involve the following vowel contraction processes: **ኡ·**\|u\|+**ኣ**\|aa\|→**ኡ·ኡ·**\|oo\|, **እ**\|a\|+**ኣ**\|aa\|→ **ኣ**\|aa\|, **እ**\|i\|+**ኣ**\|aa\| →**ኤኤ**\|ee\|. For the first person singular there is the following additional contraction involved: **መሰክ+ኩ·+ኣሁ·\|masak+ku+aahu\|→መሰኮሁ·**\|masak**koohu**\|→**መሰኮ**\|masak**koo**\|. The resulting ending -**ኮ**\|**-koo**\| is plausible considering that the first person ending **ሁ·**\|**–hu**\| has a common voiced variant **ው·**\|**–w**\|. Thus the ending **ኮ**\|**–koo**\| could easily be explained if we assume the following further contraction rule: **ኦኣ**\|**oo**\| **+ው·**\|**w**\|**→ኦኦ**\|**oo**\|.

For palatal and Ca –verbs the present perfect is formed analogously, except that the following additional contraction rule is rewired for the 3rd person masculine of palatal verbs**. ኤ \|e\|+ኣ\|aa\|→እ\|aa\|**: **ከሻን**\|kasaan\| 'he has wanted'.

3.4.3.1.5. PLUPERFECT

The pluperfect is formed by the simple past tense forms followed by the auxiliary **ናር|naar |** which is uninflected in all persons except the first person singular. Table 3.25 shows the pluperfect of verbs:

Pluperfect	Gloss
መሰኩ ናርኩ|Masakku naarku|	'I had guided'
መሰከ ናር|Masakka naar|	'You (m) had guided'
መሰክሺ ናር |Masaks naar|	'You (f) had guided'
መሰከ ናር|Masaka naar|	'He had guided etc'

Table 3.25: the pluperfect of verbs

3.4.3.1.6. NEGATIVE

Negative forms based on the perfective stem are formed with the prefix **አል|al**-|. In this way negatives can be obtained from all three tenses based on the perfective stem: **አልመሰከ|almasaka|** 'he did not guided', **አልመሰካን|almasakaan|** 'he has not guided', **አልመሰከ|almasaka naar|** 'he had not guided'.

3.4.3.2 IMPERFECTIVE (NON-PAST)

The imperfect verb in Silt'e language uses to describe non-past action. It uses prefixes and/or suffixes. As is typical for Semitic languages, some of the person markers used with imperfective stem involve both prefixes and suffixes. The affixes are shown in Table 3.26:

ይ (|y|)/እ|i-|	**/--C 1sg, 3m, 1pl, 3pl, impers.**
ይ|y-|	**/--V (for vowel ending verbs)**
ት|ti-|	**/C (for consonant ending verbs) 2m, 2f, 2pl,3f**
ት|t-|	**/--V**
The suffixes are እ|-I| 2f, ን|-na |1pl, ኡ|-u| 2pl and 3pl, |-I| impersonal	

Table 3.26: the prefixes and suffixes of imperfective verb

. The second person female suffix exerts a palatalizing influence on the last consonant of the stem, according to the following rules: **d, g →j; t, k →c; s→s, z →z, n →n, l →y.**

3.4.3.2.1. CONTINUOUS PAST

The presence of these affixes in table 3.26 is best seen in the continuous past tense, which consists of the conjugated imperfective stem followed by the past tense auxiliary.

Continuous past	Gloss
እመስክ ናርኩ|Imask naarku|	'I was guiding'
ትመስክ ናር|Timask naar|	'You(m) were guiding'
ትመስኪ ናር|Timaski naar|	'You (f) were guiding'
እመስክ ናር|Imask naar|	'He was guiding'
ትመስክ ናር|Timask naar|	'She was guiding'
እመስክነ ናር|Imaskina naar|	'We were guiding'
ትመስኩ ናር|Timasku naar|	'You (pl) were guiding'
እመስኩ ናር|Imasku naar|	'They were guiding'
እመስኪ ናር|Imaski naar|	'One was guiding'

Table 3.27: the affixes of imperfective verb in continuous past tense

3.4.3.2 .2. PRESENT /FUTURE

The present/future tense is formed by adding the present tense auxiliary (**ኣመ**፡-aammu, **ኣን**-aan, **ኣት**-aat, **ኣን**-aan,**ኣመ**፡-aammu, **ኣን**-aan, **ኣት**-aat, **ኣን**-aan) to the imperfective with its person affixes(-**ኩ**-ku,-**ክ**-ka,-**ሽ**-s, -**ክ**-ka,-**ት**-t,-**ና**-na,-**ኩመ**፡-kumu,-**ኩ**-ku,**እ**-i). Table 3.28 shows this form:

Present/future	'Gloss'
እመስካሁimaskaahu	'I guide'
ትመስካሀtimaskaaha	'You (m) guide'
ትመስኬሽtimaskees	'You (f) guide'
እመስካንimaskaan	'He guides'
ትመስካትtimaskaat	'She guides'
እመስክነንimaskinnan	'We guide'
ትመስኩመTimaskoommu	'You (pl) guide'
እመስኮንimaskoon	'They (pl) guide'
እመስኬንimaskeen	'People'

Table 3.28: the present/future form of Silt'e verb

The vowel contraction processes involved in these formations: (The second person female suffix exerts a palatalizing influence on the last consonant of the stem, according to the following rules: **d, g →j; t, k →c; s→s, z →z, n →n, l →y.)**

3.4.3.2.3. NEGATIVE

In the negative the imperfective stem takes the prefixes in table 3.29 rather than the person prefixes listed above (**ይ (y)/እi-, □y-, ት|ti-, □t-**).

እለው-|Ilaww-|	/----non- back vowel	1sg, 1pl, (1pl also እለ-/ila-)
እለው-|Ilaw-|	elsewhere	
እት-|Itti-|	/---- C	2m, 2f, 2pl, 3f
እት-|Itt-|	/--- V	
እል|Il-| imperson	/ --- V	3m, 3pl, (also 1pl),
እለ |Ila-|	elsewhere	

Table 3.29: shows the negative prefix of imperfective verb

The suffixes are the same as in the affirmative, but in the negative present/future tense the auxiliary is not added. Below the auxiliary ናር \|naar \| is added in parentheses to show the negative continuous past.

3.4.3.2.4. PRESENT/FUTURE (CONTINUOUS PAST) NEGATIVE

እለው-መስክ|Ilawmask (naar) |	እት-መስክ|Ittimask (naar)|
እት-መስኪ|Ittimaski (naar) |	እለመስክ |Ilamask (naar)|
እለው-መስክ| Ilawmask (naar)|	እለው-መስክነ|Ilawmasikina (naar)|
እት-መስኩ |Ittimasku (naar)|	እለመስኩ| Ilamasku (naar)|
እለመስኪ |Ilamaski (naar)|	

Table 3.30: shows the negative forms of present and future (past continuous tense) verbs

Note that, differing from the corresponding affirmative forms, the auxiliary in the 1st person singular here is not marked for person.

3.4.3.3 NON – ASPECTUAL

3.4.3.3.1. INFINITIVE

The infinitive is formed by suffixing **ኦት·** |–oot | to the non- aspectual stem: **ምሰኮት·** |misakoot |‘to guide’; **ክሶት·** |kisoot |‘to want’.

3.4.3.3.2. IMPERATIVE AND JUSSIVE

The jussive forms also involve a combination of prefixes and suffixes; the imperative uses only suffixes. For both imperative and jussive the suffixes are the same as those used with the imperfective stem. The prefixes are shown in Table 3.31:

ል|L| (**ኧ**|a|) -	1sg, 1pl (**የ**|ya|- is also used for 1pl)
ት·|T| (**ኧ**|a|) -	3f
ይ|Y| (**ኧ**|a|) -	3m, 3pl, impersonal

Table 3.31: the prefixes of imperative and jussive verb

The parentheses around the **a** above table indicate that this vowel is omitted preceding another vowel. Table 3.32 shows the imperative and jussive forms:

ለምሰክ|Lamsak|	let me guide	**ምሰክ**|Misak|	guide! (m)
ምሰኪ|Misaki|	‘guide (f)’	**የምሰክ**|Yamsak|	‘let him guide’
ተምሰክ|Tamsak|	‘let her guide’	**ለምሰክነ**|Lamsaknа|	**የምሰክነ**/yamsakna
‘let us guide’		**ምሰኩ**|Misaku|	‘guide! (pl)’
የምሰኩ|Yamsaku|	‘let them guide’	**የምሰኪ**|Yamsaki|	‘let one guide’
ዪመ|Yiima|	‘let him swear’	**ቱንት**|Tuunt|	‘let her close’

Table 3.32: Shows the imperative and jussive forms of a verb

3.4.3.3.3. NEGATIVE

The negative infinitive is formed from the affirmative by prefixing **አል**\|al-\|: **አልምሰኮት**\|almisakoot\| ‘to not guide’.

The suffix set for the negative imperative/jussive is the same as for the affirmative, but the prefixes are different. The prifixes are shown in Table 3.33:

አል-|al-	1sg, 1pl
አት|at-|	2m, 2f, 2pl, 3f
አይ|ay-|	3m, 3pl, (1pl), impersonal

Table 3.33: the prefix for negative imperative/jussive verb formation

Examples : **አልምሰክ**\|almisak\| ‘let me not guide’; **አትምሰክ**\|atmisak\| ‘don’t guide (m)’;**አይምሰክ** \|aymisak\| ‘let him not guide’ **አይምሰኪ**\|aymisaki \|‘let people not guide’.

3.4.3.3.4. MOODS

So far most verb forms have been given in the active mood. The passive mood is formed by prefixing ⷮ|ta-| to perfective stem forms (ⵜ|t-| after other prefix), and ⵜ|t-| to imperfective and non-aspectual forms: ⷮመሰከ|tamasaka| 'he was guided': የⵜመሰከ|yatmasaka| 'he who was guided'; እⵜመሰካን|itmakaan| 'he will be guided'.

3.4.3.3.5. MORPHOPHONOEMIC CHANGES WITH PREFIXES

Contraction processes take place with vowel and glide – initial verbs. With h- initial verbs, the ሀ/h is elided (omitted), a following short vowel is lengthened, and the prefix is realized by the allomorph it has preceding vowels. Table 3.34 shows examples of the changes with prefix:

ዬዳን|Yeedaan|	እ+ ሂዳን|i+heedaan|	'he will go'
ዮናን|Yoonaan|	እ+ ሆናን|i+hoonaan|	'he will be'
ያርታን|Yaartaan|	እ +ሀርታን|i+hartaan|	'he will swallow'

Table 3.34: the morphophonemic changes with prefixes

This rule does not apply after the affirmative jussive prefixes ለ |la-|, የ|ya-|, and ⷮ|ta :ለሂድ|lahiid| 'let me go!' የሂድ|yahiid| 'let him go !' ; ⷮሂድ|tahiid| 'let her go!'

With ወ|w-| initial verbs, regular morphophonemic changes occur only with verb forms beginning in ወ|wa-| and ወ|w-| followed by consonant: **pf +wa → oo; pf +wC →pf + uC:**

(Here the C represents the constant terms)

የ·ር዗ን|Yoordaan|	**የ·ር዗ን**+ |y+wardaan|	‘he goes down’
እለ·ር዗|Iloord|	**እለ+ወር዗**|ila+ward|	‘he does not go down’
አለ·ተ|Aloota |	**አል+ ወተ**|al+wata|	‘he did not go out’
አዩቀ|Ayuqa|	**አይ+ው·ቀ**|ay+wqa|	‘let him not hit’
አ·ቱቀ|Atuqa|	**አት· ው·ቀ** |at+wqa|	‘don’t hit’
አለ·ቆጥ·|Aluqoot|	**አል ው·ቆጥ·**|al+wqoot|	‘not to hit’

Table 3.35: Morphophonemic changes for verbs which begin with **ወ**\|wa-\| and **ው·**\|w-\|

When prefixes are added to vowel – initial verbs, the vowel, if short, is lengthened. Table 3.36 shows these phenomena:

ያ·ር዗ን|Yaarsaan|	**ይ+ አ·ር዗ን** |y+arsaan|	‘he will plough’
ታ·ድጋት·|Taadgaat|	**ት·+ አድ·ጋት·**|t+adgaat|	‘she will leave (tr)’
ቲ·ና·ት·|Tiinaat |	**ት·+ እ·ና·ት·**|t+inaat|	‘she will sleep’

Table 3.36: vowel initial verbs lengthened after they get prefixed

3.4.3.4 DERIVATION OF VERBS

It seems best to deal with derived stems from the morphological point of view, since the semantic characteristics tend to overlap and are unpredictable at times. The following derivational processes have been found:

- Preffixation of **አት·**\|at-\| plus change of thematic **አ**\| a\| to **ኤ**\|ee\|; this has a causative meaning. Thematic vowels other than **አ**\|a\| get lengthened. Verbs with long thematic vowels only add the prefix. Table 3.37 shows some examples:

ወቀ ׀waqa׀	‘hit’;	**አትቄቀ**׀atweeqa ׀	‘cause to hit’
ጀመረ ׀Jammara׀	‘begin’;	**አጄመረ**׀ajjeemmara׀	‘cause to begin’
ድነበተ׀Dinabata׀	‘be frightened ‘;	**አድኔበተ** ׀addineebata׀	‘make frightened ‘
ፎገ ׀Foge׀	‘inflate’;	**አትፎገ** ׀atfooge ׀	‘cause to inflate’
ዛሬጨ ׀Zaace׀	‘herd’;	**አዛሬጨ** ׀ azzaace ׀	‘caused to hared’
ኤፈ׀Eeffe׀	‘cover’;	**አቴፈ**׀ ateeffe׀	‘cause to cover’

Table 3.37: derivation of causative verb

This causative formation is the only derivation process that can be applied to virtually any verb, including other derived verbs**.**

- Priffixation of **አ**׀a-׀ to intransitive verbs; tends to result in a transitive meaning, but in many cases the resulting **meaning is not predictable**. Table 3.38 shows some examples of this type of derivation:

ራጀ ׀Raaje׀	‘be old’;	**አራጀ** ׀araaje׀	‘make old’
ወከበ׀Wakaba׀	‘buy’;	**አወከበ**׀awakaba׀	‘sell’
ገበ ׀Gaba׀	‘enter’;	**አገበ** ׀agaba׀	‘place in side; marry’

Table 38: derivation of transitive verbs by prrifixing **አ**׀a-׀

- Preffixation of **ተ**׀ta-׀; in most cases this passivizes underived transitive verbs. Table 3.39 shows examples of derivation of passive verbs:

ልጅ'ኘ |Ceene|	'give birth';	ተ·ልጅ'ኘ |taceene|	'be born'
ዋበ |Waaba|	'give';	ታ·በ|taaba|	'be given'
ኤወደ|Eewada|	'tell';	ቴወደ |teewada|	'be tald'

Table 3.39: derivation of passive verbs by prefixing ተ·|ta-|

- Lengthening of thematic አ |a| to ኣ |aa|; this usually accompanied by the prffixation of ተ·| ta-|; semantically it often indicates a reciprocal, iterative or very intensive process. The Prefix may be replaced by አት·|at-|, resulting in a causative meaning: ፈጀ|faje| 'finish'; ተ·ፋ·ጀ|tafaaje| 'destroy each other '; አት·ፋ·ጀ|atfaaje| 'cause people to destroy each other'. ረወተ·|Rawwata| 'run'; ተ·ራ·ወ·ተ·|taraawata| 'to run here and there'; አት·ራ·ወ·ተ |atraawata| 'cause to run here and there'.
- Reduplication: there are at least two distinct kinds of reduplicative formations: type 1 involves the reduplication of the first letter, type 2 that of the second letter by preciding with it's forth order. Semantically reduplication often indicates multiple actions – either doing something repeatedly or doing something to many objects, people etc. For a number of verbs it indicates that the activity is done 'a little', not completely or properly. Table 3.40 and Table 3.41 shows examples of type1 and type 2 reduplication respectively:

Type 1 reduplication <C1VC2 (V) →C1aaC1VC2 (v) (here C and V shows consonant and vowel respectively)

ላሀ |Laaha| 'send';	ላላሀ |laalaaha	'send many people or something to many places'
ቄረ| Qeera| 'wait';	ቃቄረ |qaaqeera| 'watch repeatedly, many times'
ልጅ'ኘ |Ceene|	'give birth';	ጫልጅ'ኘ |caaceene| ' give birth many times'

Table 3.40: Type 1 reduplication

Type two reduplication < (C1) VC2 (V) →(C1) iC2aaC2 (V)>

(Here C and V shows consonant and vowel respectively)

ፈጀ| Faje| ‘finish’;	ፍጃጀ |fijaaje| ‘finish many things’
አጀ| Aje| ‘hit’;	እጃጀ |ijaaje |‘hit many times, in many places, many people’
ኤመ |Eema| ‘slander’;	እማመ| imaama| ‘slander many times’
ኤንዘ |Eenza |‘hold’;	ቲንዛዘ| tiinzaaza| ‘hold each other’
ቀተለ |Qatala| ‘kill’	ቅታተለ|qitaatala| ‘kill many animals, people’
ጀመረ|Jammara| ‘begin’	ጅማመረ| jimaammara| ‘to start a little bit’

Table 3.41: Type two reduplication

Some verbs employ both types of reduplication, with different meanings: ፎገ\| foge\| ‘inflate’; type one reduplication: አትፋፎገ\|atfaafooge\| ‘cause to be swollen in many places’; type 2 reduplication: ፉጋገ\|fugaage\| , ፎጋገ \|fogaage\| ‘inflate many times’

With both types the original thematic vowel may also be lengthened: ክነበለ\|kinabala\| ‘return(vt)’; ክናናበለ\|kinaanaabala\| ‘turn again and again, turn over many times’ ;ፎገ\|foge\| ‘inflate’ አትፋፎገ\|atfaafooge\| ‘cause to be swollen in many places’

The reduplicated verb stems can have one or more of the derivations that a basic verb can have: causative in አት\|at-\|, passive in ተ\|ta-\|, and though much more rarely, transitive with አ\| a\|.

Table 3.42 shows examples of the three forms (reduplication, transitive and causative) together:

ሶንቹ|Sooncel ‘smell’

REPLICATED	TRANSITIVE	CAUSATIVE
ሳሶንቹ	አሳሶንቹ	አት·ሳሶንቹ
Saasoonce	asaasoonce	atssaasoonce
Many things	to smell different things	cause each other to smell something
Give a smell, something		
Gives a smell many times		

ጀመረ|Jammara| ‘begin’

REPLICATED	TRANSITIVE	CAUSATIVE
ጅ·ማመረ	አጅ·መመረ	ተጅ·ማመረ
Jimaammara	ajjimammara	tajmaammara
To start a little bit	cause to be started a little bit	to be started a little bit

Table 3.42: reduplication, transitive and causative derivation

SUMMARY

This chapter presents the Silt'e language morphology, both inflectional and derivational. The inflectional and the derivational morphologies in Silt'e language involve prefixation, infixation, prefix-suffix pair and suffixation. Hence, Silt'e words are very complex morphologically.

The language also shows some morphophonemic changes in prefixes. Hence, this also can increase the varieties of the words as well.

The complexity of the language is one of the main reasons for a language to desire a stemmer. A stemmer helps to conflate variants of words as well as to access documents in natural language text.

In Silt'e gender is determined naturally, therefore nouns do not inflected for gender but definiteness and possessive marker affixes are used according to the gender of the noun(that is naturally given) to mark it in the verb.

Verbs, nouns and adjectives are very reach morphologically to agree with person, number, and gender (not for noun) to agree with the subject in Silt'e language, the discussions are mainly focused on noun, adjective and verb morphologies.

Silt'e language has two types of reduplication and discussed in this chapter as type1 reduplication and type 2 reduplication. Type1 with first consonant reduplication and the second (type 2) indicated by the second consonant reduplication.

The forms of Silt'e verb such as perfect (past and completed actions), imperfect, jussive, and infinitive are discussed. Its derivational characteristics: causative, replicated transitive causative and passivasing of verbs also discussed.

The next chapter presents the development of a stemmer to conflate variants of Silt'e words with respect to concepts in this chapter.

CHAPTER FOUR

A STEMMING ALGORITHM DESIGN AND IMPLEMENTATION (PROTO TYPE) FOR SILT'E LANGUAGE TEXT

4.1 INTRODUCTION

A morphology part of the Silt'e language has been presented (reviewed) in chapter three. It has been shown that the main word formation process in Silt'e is done through affixation. As described before the parts of affix are: - prefix, suffix, prefix-suffix pair and reduplication (type1 and type2 in Silt'e language). These affixes are used for inflecting and deriving words. Nouns are inflected for definiteness, number, and person. Verbs are inflected for gender, number, person, aspect and tense. Like other Semitic languages, Silt'e experiences complex morphological structure that resulted in very large variants of a word. In designing retrieval systems and other natural language processing systems for the language, reducing these variants into one form improves the performance of the systems. This can be achieved by a conflation technique, which is usually stemming. The main purpose of a stemmer is reducing different variants of words in to their standard form (root). Thus, the main reason for this chapter is designing (developing) a stemming algorithm for the language. Therefore, the following part presents development of a stemming algorithm for the language. The compilation of stop words and affixes and evaluation of the stemmer has been presented as well.

4.2THE CORPUS

The researcher has utilized different sources of sample text for the development of the stemmer and the experiments. Since there is no document collection for any of Ethiopian languages available likes TREC and CLEF collections for the English language, the researcher took documents from different sources and organized them into one collection. The corpus consists of 10,922 tokens or 5,288 distinct words. The document has been used for stop word list collection; affix collection and testing the algorithm. The test data was compiled from the document randomly to check the stemmer from different angles of varieties of words.

4.3WORD DISTRIBUTION OF SILT'E

Word distribution in sample text documents of a language helps to study language's behavior, and this distribution can be shown using word-ratio (number of distinct words to total number of words). This helps to show how many words are morphologically distributed within a document [9]. Table 4.1 shows the word ratio of sample document for Silt'e language.

Name	Description	Word tokens	Word types (Distinct)	Word ratio (distinct to total word)
SampleDocument	Different books of Silt'e in academic area and other: • Text book from grade1 to 8 • Text book from grade 9 to 10 • Ficsions • peoms	10,922	5,288	48.42

Table 4.1: Number of words and their distributions to Silt'e sample text data

Sample document in Table 4.1 used for compilation of stop words, affixes (prefix and suffix) list and for testing the developed algorithm.

To compare the sample sets of Silt'e word distribution with other languages such as Ge'ez [44], Tigrigna [40], and Amharic [42], Table 4.2: show sample text of (Text1) with 1,518 numbers of words was taken randomly from the SampleDocument in Table 4.1. The word ratio for sample text of Geez, Tigrigna, Arabic and English languages has been adopted from [44].

language	Text	Total number of words	Distinct words	Word ratio (distinct to total word)
Silt'e	Text 1	1518	823	54.22 %
Ge'ez	Lukas	1,866	1,064	57.02 %
Tigrigna	Text1	1,632	918	56.25 %
Arabic	Text1	1,600	902	56.38 %
English	Text1	1,600	621	38.81%

Table 4.2: comparison of word distribution ratio of the sample data set among different languages

The higher value of the word ratio corresponds to more distinict words in a text and vise versa [88]. The word ratios obtained for Silt'e sample text Text1is some what similar to Ge'ez, Tigrigna, Amharic and Arabic texts as shown in Table 4.2. However, it is absolutely different from English text. This similarity among Silt'e, Ge'ez, Amharic, Tigrigna and Arabic might be due to the fact that each is in Semitic language groups. As Table 4.2 shows numbers of unique words found in Silt'e document is similar with Semitic languages and it is larger compared with the English language distinict words. Hence, Silt'e language has comparable distinct words with morphologically very complex languages when its word distribution is compared with others based on their sample data sets. And when it is compared with English it is morphologically very complex.

4.4. DOCUMENT (MORPHOLOGICAL) PREPROCESSING

The stream of characters in natural language text must be broken up in to distinct meaningful units before any language processing can be performed. Preprocessing is the most important part of all text processing. 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 [50]. In the preprocessing stage this study addresses tokenization, normalization and stop word removal.

4.4.1. NORMALIZATION

All punctuation marks, control characters, numbers and special characters are removed from the text before the data is processed.

In contrast to Amharic and Tigrigna, Silt'e uses distinct 26 Ethiopic alphabets (in Silt'e language writing system different letters that has similar sound, like ሐ, ኀ, ኸ, ሠ, ዐ are not used, instead only ሀ, ኣ, ሰ are used). But, in some document of the real life, people experienced writing of these similar sound characters which are used in Amharic and Tigrigna writing systems. Even though these symbols sound the same, in Silt'e language they are not part of the alphabet. But in case people use them they must be changed to alphabets that are used in Silt'e, that have similar sound form**.** Therefore, for the sake of flexibility the system will handle this kind of situations as they appear in the documents.

4.4.2. TOKENIZATION

In this study, words are taken as tokens. Since all punctuation marks have been converted to spaces, 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 tokenization processes.

4.5. COMPILATION OF THE STOP WORD LIST

The stopword list was compiled from the Sample document in Table 4.1 by collecting the most frequently occurring words. Frequency of words in the document was generated by using python program. Table 4.3 lists the top 54 frequent items from the document. As can be seen from the Table 4.3, the stopword list consists of prepositions (ኡስጥ|in|), demonstrative adjectives (ሂተይ|this|, the masculine singular), articles (|እታይ|the, masculine singular, |እተቲ| the, feminine singular) and conjunctions (suchas |ጊንመ| but, |ዋ|and). As we can see in Table 4.3 there are words with high frequency but are not stop words (such as አሰነት, ቅሬታቶ, ገረድነት, የአዳ, አደ, አህላቅ, ጋር, ባሎት, የገረድ, ወክት). As indicated in chapter three, function words such as pripositions, conjunctions and articles in Silt'e exist affixed to words. For this reason the

frequency of function words in Silt'e is low. Moreover the stopwords in Silt'e can exist in different forms by concatenating affixes. Therefore this also decreases their frequencies. Table 4.4 shows sample list of words that have low frequencies but are stopwords. See the complet list in appendex IX. Because of this reason, the use of frequency alone did not help to generate all the stopwords and the researcher was used research paper that talks about the morphology of Silt'e language to collect stopwords from the SampleDocument. Many function words were also added manually to the stopword list by consulting dictionary [53] and research paper [52]. Table 4.5 shows the list of sample stop words that are used in the development of the algorithm. Generally, 184 stopwords are used to develop the stemmer.

No	Word	frequency	No	Word	Frequency
1	ደር	73	41	አደ	17
2	ሰብ	71	42	ሀድ	16
3	አነግነ	71	43	ጀዋብ	16
4	ኤት፡	62	44	ኮሎ	16
5	ወክት፡	57	45	ጉት፡	16
6	ደረሳሶ	57	46	ሚሽ	15
7	ወልድ	44	47	ያሉ	15
8	ኡስጥ	43	48	ዮነከ	15
9	ቅሬታቶ	39	49	ገረድነት፡	15
10	ባድ	39			
11	ሃለት፡	38			
12	ሱልቸ፡	36			

13	የገረድ	35			
14	አሰነት፡	35			
15	ሉላሉሌ	34			
16	ያለይ	29			
17	ያለይመ፡	29			
18	የአዳ	29			
19	አድ	29			
20	ገረድ	28			
21	አህላቅ	27			
22	ዋ	26			
23	ወግ	25			
24	ግዝ	24			
25	ጋር	24			
26	የሰብ	21			
27	ባለ	21			
28	የውን	21			
29	ስንቅ	20			
30	ገነገናም	20			
31	ገዳ	20			
32	ባሎት፡	20			
33	ጃንጎ	19			
34	ድራመ	19			
35	ያቀት፡ሎን	18			
36	ያቴራን	18			

37	**ዮነ**	**18**			
38	**በ·ር**	17			
39	**በቅራ·ቲ**	17			
40	**ሬር**	17			

Table 4.3: The most frequent words from the SampleDocument

Words with lower frequencies but are stopwords

Word frequency	Word frequency	Word frequency	Word frequency
ገናኒሚ 2	ልቶን 3	ኢነ 2	ጊዘቸ 2
ጃንሎ 3	ቦንት 2	ኢንኩ 2	ሂንኩ 2
የሰሀደደይ 2	ያፍቱቱ 2	ኢነኮ 2	ዩሃን 2
ለገጊኒሞ 2	እቱታን 1	ባታይ 3	በገጊሽ 2
አደደሞ 2	እነይ 2	ገነግነ 2	
ሂተይ 2	እታይንገ 2	ሎኔት 2	ለሆነ 2
ጊንመ 2	ሊሊ 2	ሂንኩምከ 2	ሂንኩም 5
አደኒመ 2	ዮናነይ 2	ታት 2	ዩኑ 2

Table 4.4: words with lower frequencies but are stop words

አዩ	አልተቱ	እሊ	እነይ
እታይ	እትታይ	አሊ	በልዳሌ
እተቱ	እትተቱ	እነይ	አልቀሬ
አታይ	አትታይ	አነይ	ደር
አተቱ	አትተቱ	እቲ	ኤት
ዩታይ	እብታይ	አቲ	ፈሬ
ዩተቱ	አብተቱ	እቢ	ፎኖ
ያታይ	እብታይ	አቢ	ግነ
ያተቱ	እብተቱ	ሱር	ቀደ
እልታይ	አብታይ	አድ	
እልተቱ	አብተቱ	አድአድ	
አልተይ			

Table 4.5: sample list of stop words

4.6. COMPILATION OF AFFIXES

In Silt'e affixes consist of three different types, which are the prefix, suffix, and infix. Unlike English stemmers which work quiet well just by removing suffixes and few prefixes to obtain the stems, an effective and powerful Silt'e stemmer not only must be able to remove suffixes, but also prefixes, and infixes as well. Without removing all this affixes, the stemmer cannot be effectively used to stem Silt'e documents.

4.6.1. COMPILATION OF PREFIXES

A set of prefixes that are used to develop the algorithm is collected from the SampleDocument by using Evan Gut's research paper "Concise grammar of Silt'e" [53] as a guide and from the paper

itself. Generally 36 prefixes are used to develop the stemmer. Table 4.6 shows some of the prefixes collected for the development of the algorithm

እለው	ላ
እለው	ቃ
በሰ	ጫ
አይ	አ
እት	ለ
እት	የ
እል	ይ
እለ	እ
እለ	ተ
አል	ት
አት	

Table 4.6: sample list of prefixes

4.6.2. COMPILATION OF SUFFIXES

To collect a lis of suffixes from the SampleDocument the researcher uses Evan Gut's research paper "Concise grammar of Silt'ee" [53] as a guide and collected from the paper itself. A total of 68 suffixes are used for the purpose of developing the stemmer. Table 4.7 shows samples from a list of suffixes used in the algorithm.

አመ፡	ኩሞመ፡	እ
ኩመ፡	አን	ይ
ኒመ፡	ኤን	ሚ
ኣሀ፡	አኘ	ዋ
ኣው፡	አተኘ	ሶ
ኣሀ	ኣንቑ	ዬን
ኣሸ	ኣም	ኒመ፡
ኣነ	ኣታም	
ኣት፡	አታም	
ኣመ፡	እሎ	

Table 4.7: sample suffixes

4.7. THE PROPOSED STEMMER

In this research the data have been represented in Unicode and the stemmer accepts Unicode data directly without transliterating it to Latin form. Therefore, the length is used to represent the size of the word while developing the algorithm. The minimum length for a meaning full Silt'e word is two. Most words in Silt'e language have length of two [53]. Thus, the minimum length of word size has been used in the algorithms based on this concept.

Techniques developed for English and Semitic languages such as Amharic, Tigrigna and Geez have been studied. Some of the techniques used in these algorithms were incorporated in developing the Silt'e stemmer. The algorithm uses iterative approach but it removes the longest affix first.

As indicated in chapter three, the process of inflection and derivation in Silt'e is done through one or the combination of the following processes.

- Pure external affix (without modifying the stem)
- External affix and with modification of the letter(s) of the stem at the beginning, end or other positions
- Pattern change (insertion or deletion of consonant and vowels)
- Reduplication

Accordingly, these characteristics were taken in to consideration in developing the stemmer.

Of the two approaches (context free and context sensitive), a context sensitive approach was considered appropriate, as Silt'e is morphologically complex language (See Chapter three).

In the stemmer developed, three steps were used for the purpose of removing affixes. The purpose of each step is given as follows:

The first step removes prefixes. In removing a prefix, checking a match in stop word list, prefix list and counting length of the word and checking contexts sensitive conditions is done. If the word gets match in the prefix list and satisfies the conditions for context sensitive, it will be returned without removing the prefix, otherwise the prefix striping process takes place.

In the second step the removal of suffix is done. First the word is checked against stop word list. If there is no match in the stop word list then the length of the word will be checked. If it is greater than two then the suffix list file will be opened and checked if there is a match of suffix with word. If the word has a match and doesn't satisfy the context sensitive conditions the suffix striping process will be performed. Otherwise no suffix will be removed.

The last or the third step is used to stem reduplication of letters from the word. This step deal with type 1 and type 2 reduplication in Silt'e language. First the length of the word is checked. If length greater than two then the program checks existence of reduplication in a word. If the word holds reduplication of the first letter then the first reduplicated letter will be removed from the word. Otherwise the reduplication of the second letter will be cheked if exist then get stripped. If the word doesn't have any kind of reduplication then it will returned as it is.

Except the removal of reduplication, after each step is applied, the word is checked against some context sensitive conditions. This is because to avoied the removal of non genuen affixes.

The proposed stemmer gets the word and then checks if it is in stop word. If a match is found the stemmer starts again and takes the next word if end of file not reached else the word will be given for the next process and the length of the word will be counted. If length of the word is less than three it will not be stemmed and get written to the stemmed file otherwise the word will be given the affix removal process to strip the appropriate affixes. The algorithm checks each time the word against the stop word after prefix and suffix stripping process. The recoding and context sensitive checked and based on the conditions the appropriate action will be taken. Flow chart of the stemmer algorithm shown in figure 1.

4.8. RECODING AND CONTEXT SENSITIVE RULES

A context-free stemmer is one that removes strings with out consideration of the remaining stem, and can thus remove strings that are similar to, but that actually are not, genune affixes. English examples of such behavior include removing "re-"from "regular" or "-al" from "metal" [86]. And equally poor results are obtained if context free is adoped for Silt'e. For example **አቦት**|abot| 'father', **አረሰ**|arasa| 'plough', have a leading string**አ** |a| which is in the list of prefix removing this leading string result in **ቦት** |bot|, **ረሰ** |rasa| doesn't have any relation to the correct stem. There are many examples in Silt'e and it was hence decided that a context-sensitive approach sould be adopted, with the affix removal is being controlled by two action cods and three conditions. The two types of context sensitive actions are:

Action 1: don't perform any affix removal

Action 2: remove affix

4.8.1. RECODING RULS

Recoding1: The word ends in the letter of six order form: this is to recode the change in order form at the end.

Recoding 2: The word begins with affix: If character '**ዮ**' are part of identified stem word change it to **የወ**

4.8.2. CONTEXT SENSITIVE CONDITIONS

Condtion 1: The affix followed by: order of letters. This is to avoid the removal of non-genune affixes.

Condition 2: The letter to be removed is reduplicated and in the first order: take action 1, this is to avoid removing non-genune type 1 reduplication like ጠጠረ |tatara|

Condition 3: The lenth of word to be considered: if the length of the word is less than two, take action 1 this is to maintain the minimum stem length of the word in the language

Certain amount of terminal character recoding can occur in Silt'e. Examples of characters that can be recoded (converted) are ሻ, ሸ, ቾ, አድ, ኩ. . A recoding routine was thus included to handle such occurrences. Table 4.8 shows some recoding examples:

Ending	Recode	Example
ሻ	ት	አቦሻ to አቦት
ት	ደ	ሄት to ሄደ
ሸ	ድ	ባሸ to ባድ
ቾ	ድ	ገራቾ to ገራድ

Table 4.8: examples of recoding

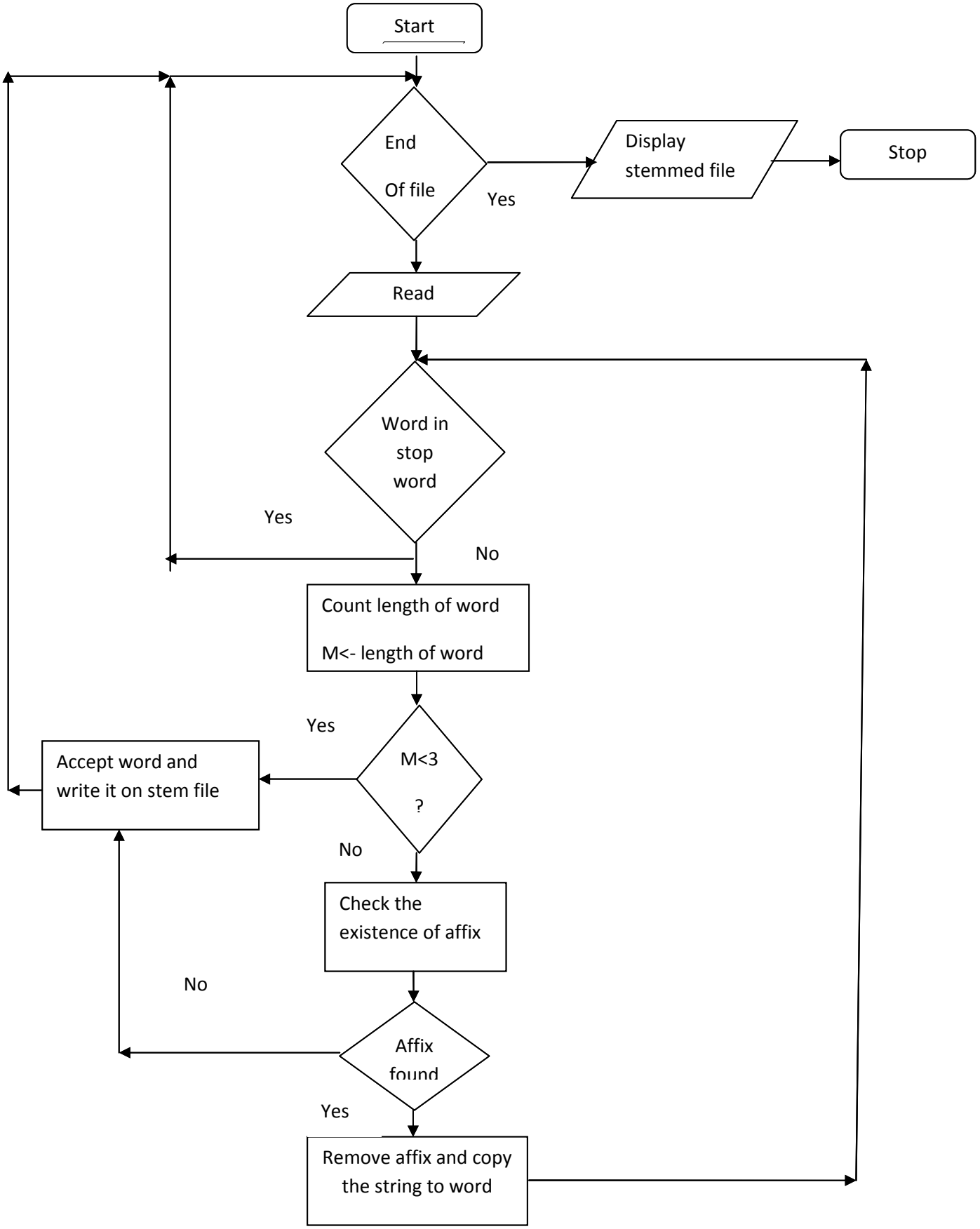


Figure 4.1: Flow chart for the general affix removal algorithm

4.9. COMPONENTS OF THE STEMMER

The stemmer processes prefix striping and suffix stripping based on the affix lists stored in the affix list files. As the affix found, the word will be checked with the context sensitive conditions if the conditions are not fullfilled the affix removal takes place. There are also affixes that are striped using iterative approach with rules of the language, for instance, lists of characters such as በ|ba|, የ |ya|, ተ|ta| or ሰ|sa| can occur by concatenating each others as prefix of words. In removing of affixes, the sequences that are done in this work are : <checking word length>, <prefix checking>,[prefix removing], <suffix checking>,[suffix removing],<checking letter reduplication type1 and 2>,[removing the duplication type1 or two]. In each activity there are stemmed word length checkers, stop word checkers and the recoding and context sensitive rules to result in good stemmed words as possible.

4.9.1 THE AFFIX- REMOVAL TECHNIQUES

Before this technique is applied, some common stop words from sample text are removed by matching with stop word lists. In the process of affix removals, the length of each word is also checked.

The stemming process is done whenever length of word is greater than two. This is because the minimum length of words in Silt'e is two [53]. The stemmed word, after affixes are removed is also checked with list of stop words and if it is part of stop words, it will not go to the affix removal process or will not be part of the stemmed file.

Basically, two actions are taken in the stemming processes.

Action 1: do not remove any affix

Action 2: remove the concerned affix

To take any one of the above actions, there are rules (conditions) that are used to check whether the word is context sensitive or not and apply action 1 or action 2. The conditions are:

Condition 1: after getting the assumed prefix or suffix, if length of word is not more than two or fullfil context sensitive rule take action 1.

Condition 2: if part of the assumed affix found and the length of a word is greater than two and if the word is not fullfill context sensitive rule take action2.

In the stemming process, based on the above condtions the appropriate action is taken.

4.9.1.2 PREFIX STRIPING

The procedure takes a worlopd and checks the existence of a true prefix, and removes it whenever the condition is fulfilled. For example a verb which begin with **ኣ** |a|, and if the word remaining is more than two it will strip the prefix after checking the context sensitive rules. If there is no condition to be fulfilled the prefix striped other wise the word returned. For instance the word **ኣቦት·** |abot| will not go under prefixing process because, **ኣ**|a| is not a true prefix for **ኣቦት·** |abot| as the remaining string **ቦት·** |bot| has no meaning in Silt'e language. Where as **ኣ**|a| is true prefix for **ኣበለ** |abala|. Hence undergo in the stripping process and **ኣ** |a| will be removed and the remaining stem **በለ**|bala|,'eat' will be returned. Table 9 shows the prefix striping stemmer algorithm:

1 Get word

2 Read stop word file and check if there is match with word

If word exist in the stop word list then

Go to step 5

Else

Find the length of the word

3 If if length of word < 3 then return word and Go to step 5

Else :

If length of (word)<=(Maximum Prifix length +2) then

assign the first length (word)-2 characters to a temporary

prefix variable

Else

assign the first Maximum Prifix length to a temporary prefix variable

4 Read the prefix list file and check if match found with a temporary prefix variable:

If match found then ckeck conditions:

If context sensitive rule fulfilled:

Return word and Go to step 5

Elese :

Remove prefix and Go to step 2

Else

If length(a temporary prefix variable) >1 then:

assign length(a temporary prefix variable)-1 characters to a temporary prefix variable

And go to step 4

Else

Return word and Go to step 5

5 If end of file not reached Go to step 1

Else:

Stop processing

Table 4.9: prefix striping stemming algorithm

1 Get word

2 Read stop word file and check if there is match with word

If word exist in the stop word list then

Go to step 5

Else

Find the length of the word

3 If if length of word < 3 then return word and Go to step 5

Else :

If length of (word)<=(Maximum suffix length+2) then

assign the last length (word)-2 characters to a temporary suffix

Else

assign the first Maximum suffix length characters to a temporary suffix

4 Read the suffix list file and check if match found with temporary suffix :

If match found then ckeck conditions:

If context sensitive rule or recoding rule fulfilled:

Take the appropriate recoding

Return word and Go to step 5

Elese :

Remve prefix and Go to step 2

Else

If length(temporary suffix) >1 then:

assign length(temporary suffix)-1 characters to temporary suffix

And go to step 4

Else

Return word and Go to step 5

5 If end of file not reached Go to step 1

Else:

Stop processing

Table 4.10: Suffix Stripping Stemming Algorithm

4.9.2 SUFFIX STRIPPING

The procedure takes a word and checks the existence of suffix match in the suffix list. If match found then will cheeked against the context sensitive conditions and based on the conditions the appropriate action will be taken. The process is almost the same with the prefix striping except the stripping and the affixes are on the right side of the word. Table 4.10 shows the suffix striping stemmer algorithm:

4.10. LETTER REDUPLICATIO STRIPING

4.10.1. TYPE 1 REDUPLICATION

This procedure deals with reduplication of the first letter of the word, i.e.ላላሀ|laalaaha|. The procedure takes as an input the output of the previous procedure. It checks the existence of the reduplication and removes the first letter of the word if it is reduplicated and the letter is not in the first order of the alphabet. Otherwise the procedure returns the word as it is. For instance the word ላላሀ|laalaaha| contains reduplication and the first letter will be removed and resulted in word ላሀ|laaha|, 'He send'. The procedure checks the alphabetical order of the letter to avoed the removal of the reduplicated letter if it is like Amharic word ጠጠረ|tatara|.

Table 4.11 shows type 1 reduplication stemming algorithm:

1Get Word

2 Find the length of the word

3 If length of word < 3 then:

return word

Go to step4

Else:

If first letter of word is the same to second letter of word then:

If the first and the second letters in first order form then:

Return word

Go to step 4

Else :

Remove the first letter

return the stripped word

Go to step 4

Else:

If the first and second letters are the forth and the fifth orders of the same alphabet:

Remove the first letter

return the stripped word

Go to step 4

Else: return word and Go to step 4

4 If end of file not reached Go to step 1

Else :

Stop processing

Table 4.11: type1 reduplication stripping stemmer algorithm

4.10.3. TYPE 2 REDUPLICATION

This procedure deals with the reduplication of the second letter (infact by reduplicating it self to its fourth form), i.e. ጅማመረ |ji**m**aa**m**aral. It checks the existence of the reduplication and removes the second letter of the word if it is reduplicated and returns the stripped word, otherwise the procedure returns the word as a final output. For example, the word ጅማመረ |jimaamaral| contain reduplication and the second letter will be removed and result in word ጅመረ|jamaral|. The procedure will not deal with words like አበበ |ababa|, in Amharic because the reduplication in the word አበበ |ababa| is not in the fourth and first order of the alphabet, i.e letter በ|ba| and በ|ba| are both in first order of the alphabet. Table 4.12 shows type 2 reduplication stemming algorithm.

1Get Word

2 Find the length of the word

3 If length of word < 3 then:

return word

Go to step4

Else:

If second and third letter of word are the fourth and the first orders of the same alphabet then :

Remove the second letter

return the stripped word

Go to step 4

Else:

Return word

Go to step 4

4 If end of file not reached Go to step 1

Else :

Stop processing

Table 4.12: type 2 reduplication stripping stemmer algorithm

4.11. IMPLEMENTATION OF THE STEMMER

The Silt'e stemmer implemented as a sequential program using python programming language. The algorithm implemented is the longest match first and the lists of affixes are checked against the word. It removes affixes iteratively until the entire affixes are removed. Table [4.**13**, 4.**14**, 4.**15,** and 4.**16**] show sample words prefix stripping; suffix stripping, letter reduplication (type 1) and letter reduplication (type 2) results of the Silt'e stemming algorithm:

No.	Prefixes to be removed	The word to be stemmed	Word after stemming
1	**በ**	**በደውሰይ**	**ደውሰይ**
2	**የ**	**የሚሸትክ**	**ሚሸትክ**
3	**የ**	**የስልጤ**	**ስልጤ**
4	**በ**	**በቡርደክ**	**ቡርደክ**
5	**ተ**	**ተቄሴ**	**ቄሴ**

Table 4.13: Sample of prefix striping

No.	word before stemming	suffixes to be removed	Word after stemming
	ደውሰይ	**ይ**	**ደውሰ**
	ሚሸትክ	**ክ**	**ሚሸት**
	ቡርደክ	**ክ**	**ቡርደ**
	በቶጵያ	**በ**	**ቶጵያ**
	መ·ት·ቸ	**ቸ**	**መ·ት·**
	ሱ·መኒመ·	**ኒመ·**	**ሱ·መ**

Table 4.14: Sample of suffix striping

Reduplication to be removed	The word to be stemmed	Word after stemming
ና	**ናናገ**	**ናገ**

ላ	ላላሀ	ላሀ
ላ	ላላሰ	ላሰ
ሳ	ሳሳደ	ሳደ

Table 4.15: Sample of type1 the reduplication striping

Reduplication to be removed	The word to be stemmed	Word after stemming
ባ	ድባበለ	ደበለ
ቃ	ንቃቀለ	ነቀለ
ቻ	ንቻቸ	ነቸ

Table 4.16: Sample of type2 the reduplication striping

4.12. EXPERMENTS AND DISCUSSIONS

4.12.1. 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: over stemming and under stemming. When a term is over stemmed, too much of it is removed. Over stemming can cause unrelated terms to be conflated. Under stemming is the removal of too little of a term. Under stemming will prevent related terms from being conflated. To evaluate the performance of the stemmer, mannual counting technique was used. This helps to compare number of errors that are not conflated correctly with the correct one. The stemmer was tested the sample text1 that is taken randomly from the sample text document. Sample text1 contains 1,486 words. During the stemming process two types of errors are observed. These are under stemming and over stemming. The stemed result of the sample text is 966. Out of these words 4.24% (41) words were under stemed, 10.04 % (97) words were overstemed. Totally the stemmer generates 14.28 % (138 words) stemming error. As a result, the accuracy of the stemmer becomes 85.71 %. See some errors of the stemmer in Table 4.17, Table 4.18 and table 4.19, shows the summarized result of the evaluation.

Words	Resulting stem	Expected stem	Error type
አይነትቸ	ነት	አይነት	Over stemming
ኡመት	ኡመ	ኡመት	Over stemming
	ቡርዳሙ	ቡርዳ	Under stemming
□□□□□	ደም	አበድ	Over stemmingr

□□□	ወቅ	ወቅት፡	Over stemming
ብላት፡ቸ	ብላት፡	ብላ	Under stemming
ስብነት፡	ስብነ	ስብ	Under stemming

Table 4.17: examples of stemming errors

Table 4.17 shows sample errors observed during the stemming process of silte stemmer.

4.12.2. THE RESULTS

Table 4.18 and table 4.19, shows the summarized result of the evaluation.

In table 4.18 column name Data set contains the sample text used in the evaluation process, column words show the number of words in the sample text1, the figure 966 resulted after the sample tex1 exeperianced the stemming process, out of 966 words correctly stemmed are 828 hence resulted in the accuracy of 85.71%.

Data set	words	stems	Correct stems	Percentage (%)
Sample text1	**1486**	966	828	85.71

Table 4.18: correct stem

Table 4.19: illustrates the distribution of errors for the incorrectly stemmed words in Sampletext 1.

Out of the stemmed words 14.28 % (138) words were incorrectly stemmed. Over stamming account for 10.04 % (97) words and under stamming 4.24 % (41) words.

Data set	Errors (%)	Over stemming (%)	Under stemming (%)
Sample text1	14.28	10.04	4.24

Table 4.19: distribution of errors

The errors that observed from the stemmer could be due to the following reasons:

1. Because of the complexity of the language, it was difficult to come up with complete list of affixes at a time.

2. More context sensitive rules are required based on the detailed study of the morphology of the language

WORD COMPRESSION RATIO

Sample text1=1,486 words. The number words of the test data after stemming also counted and are 966.

Stemed=966

The stemmer is also evaluated with percentage of compression. For calculating the compresiton rate(C), the expression used by [9] was used. The expression is shown as follow:

$$C = \frac{100 * (W - S)}{W}$$

Where W number of total words

S number of stemed words from W

Sizeof the data=1,486

Number of stems=966

Hence, the percentage of compression for Silt'e text based on the training text for

Stemmer becomes 100 *(1,486-966/1,486) =34.99%

The compration ratio is 34.99% for the stemmed words. It reduces the sample text by 34.99%.

SUMMARY

When the stemming process is done, over stemming and under stemming problems is observed.

For the experiment, 1486 words, which collected randomly from the available text document sources, have been used. From the data the performance of the stemmer is evaluated and the result shows 85.71 % accuracy. From this data set of words, 34.99% of compression of stemmed words found. The total errors observed from the experiment were 14.28%.

The stemmer conflates derivational and inflectional affixes. It does not conflate irregular and compound words.

The next chapter presents conclusions of findings and recommendations for future research.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATION

5.1 CONCLUSIONS

Silt'e is one of the Semitic languages. These languages have common grammatical system based on the root pattern structure. The main word formation process in Silt'e is done through affixation. Silt'e uses prefix, infix and suffix. It also uses reduplication of part of a word, which is first constant reduplication and the second constant reduplication of a word.

The process of adding one affixe to another can result in relatively long words, which often contain an amount of Semantic information equivalent to a whole English phrase, clause or sentence. A Silt'e word can give rise to a very large number of variants, with a consequent need for effective conflation procedures if high recall is to be achieved in searches of Silt'e text data bases. In this research, the possibility of developing stemming algorithm for the language was investigated.

The quantitative analysis done on the sample text showed that words are distributed through the texts in their morphological variants.

Stemmers developed for other languages could not be applied for this language because of the morphological complexity and difference in features of the language as discussed in chapter three. However, commonly used methods of stemmers such as using affix dictionary, stop word list and context sensitive rules are employed. Also some techniques are adopted from Porter and Nega in developing the stemmer. Stripping suffix is not enough to conflate variant words of Silt'e to one form. Hence the stemmer also includes procedures to remove prefix and reduplication of letters.

To evaluate the stemmer developed in this study, test data of size 1486 words were selected randomly from the sample texts. The experiment showed that the stemmer performs at accuracy of 85.71 % and reduce dictionary size by is 34.99% for stems. This shows that using a stemmer for Silt'e brings a significant reduction in dictionary size as a result of conflating variant words

to the same stem. The results obtained from the experiment are promising and using the stemmer in IR system of the language could improve the performance of the system.

The stemmer conflates only inflectional and derivational affixes. It does not conflate compounding and irregular forms. Infix and prefix-suffix pair also not covered in this research. The resean is that it is possible to put prefix part to prefix list and suffix part to suffix list to strip the prefix-suffix pair of the word. As far as the infix part of the language concerned it is difficult to handle it within short pered of time. There are different ordering possibilities of applying the procedures. In this experiment the stripping order was prefix, suffix and letter reduplication. Other possibilities could not be tested due to limitation in time.

5.2 RECOMMENDATIONS

Although encouraging result has been obtained in this research work, the following recommendations are identified for further work in order to make the result useful in operational retrieval environment:

This research showed the possibility of developing a stemmer to conflate word variants of Silt'e language, which has complex morphology and where a word can have very large variants. However the study was based on a limited size of sample texts and not tested in IR environment because of the time constraint. The operation of the algorithm depends on the order of the striping procedure, despite the fact that it is not clear in what order the procedure should be applied to an input word to obtain correct stem. The ordering could be based on arbitrary criteria or on linguistic analysis. Due to limitation in time the merits of the different possible ordering could not be studied in this experiment. Studying the effect of ordering the stripping procedure on the performance of the stemmer and selecting the best possible ordering.

Evaluating the stemmer on text collection of large size collected from different sources. This is because large size sample can represent the characterstics of the language more than small size sample. Therefore, the accuracy of the stemmer can better be checked this way.

This stemming algotithm developed based on Unicode data, it is recommended to use transliterated data to make it more efficient.

Evaluating the stemmer in IR environment (if developed in the future) to measure its performance in actual retrieval session.

One can add more context-sensitive and recoding rules in order to increase the accuracy of this stemmer.

Moreover, by doing additional research on the language, this work can help to develop application tools such as spell checker, parser, thesaurus and dictionary.

REFERENCES

[1] Gerard Salton. Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer, Addison-Wesley, Reading, MA, 1989.

[2] Christian Jacquemin, Guessing morphology from terms and corpora, In Belkin et al [1], pages 156-165.

[3] Wessel Kraaij. Viewing stemming as recall enhancement, In Hans-Peter Frei, Donna Harman, Peter Schauble and Ross Wilkinson(editors), Proceedings of the 19th Annual International ACM-SIGIR Conference on Re-search and Development in Information Retrieval, pages 40-48, Zurich, Switzerland, 18-22 August 1996. ACM

[4] Robert Krovetz. Viewing morphology as an inference process, In Robert Korfhage, Edie Rasmussen and Peter Willett (editors),Proceedings of the 16th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, pages191-202, Pittsburg, U.S.A., June 27 - July 11993. ACM

[5] J. B. Lovins. Development of a stemming algorithm, Mechanical Translation and Computation, Volume 11, Number 1-2, pages 22-31, 1968.

[6] M. Porter, An algorithm for suxffix stripping Program, Volume 14, Number 3, pages 130-137, 1980.

[7] Evelyne Tzoukermann, Judith L. Klavans and Christian Jacquemin, Effective use of natural language processing techniques for automatic conation of multi-word terms: the role of derivational morphology, part of speech tagging, and shallow parsing. In Belkin et al [1], pages 148-155.

[8] Jinxi Xu and W.Bruce Croft, Corpus-based stemming using co-occurrence of word variants.ACM Transactions on Information Systems, Volume 16, Number 1, pages 61-81, January1998.

[9] Donna Harman, How effective is suffixing?, Journal of the American Society for Information Science, Volume 42, Number 1, pages 7-15, 1991.

[10] David Hull, Stemming algorithms: a case study for detailed evaluation. Journal of the American Society for Information Science, Volume 47, Number 1, pages 70-84, 1996.

[11] Chris D. Paice, An evaluation method for stemming algorithms, In Croft and van Rijsbergen, pages 42:50.

[12] Lenneon, M. Peirce, D. Tarry, B. and Willett, P. "An evaluation of conflation algorithms for information retrieval." Jornal of information scence, 3,177-183, 1981 .

[13] Wakshum Mekonen, "development of Stemming algorithm for Afaan Oromo Text", M.Sc. Theses, Addis abeba University, 2000.

[14] Tesfi Tewedrios, "Word formation in Tigrinya" M.S.c Theses, Addis Abeba Unversity, 1993.

[15] Savoy J. "Stemming of French words Based on grammatical Categories "Journal of American society for information science, 44(1) 1-9, 1993.

[16] Ahmed F., et al, Experiments with stemming algorithm for Malay words, Journal of the American Society for Information Science, 47(12), 909-918, 1996.

[17] Al-Kharashi, I.A. and Evens, M.W, "Comparing Words, Stems and Roots as Index Terms in an Arabic Information Retrieval Systems "Journal of the American Society for information science, 45(8), 546-560, 1994).

[18] Popovic M. and Willett P. , "The effectiveness of Stemming for Natural Language Access to Slovin Textual Data" Jornal of American society for Information sciences, 43(5), 391-395, 1992.

[19] Ekmekcioglu C.F., Lynch M.F and Willett P. ,"Stemming and N-gram Matching for term conflation in Turkish Texests", 1996.

[20] B.L. Narayan and Sankark.Pal, Distribution Based Stemmer refinement Machine intelligence unit, Indian, Statstical, Institute, Calcutta-700108, India, Pp.1-6.

[21] David A. Hull, Stemming Algorithms-A case study for Detailed Evaluation, Pp.1-27, 1995

[22] S.Srinivasan and ZP.Thambidurai, Stans Algorithm for Root Word Stemming, Information Technology Jornal 5(4): 685-688, India, 2006.

[23] Abdule Basit M., Gowende R., Husen .A, and Abduselam M., "A Hibrid Method of Stemming for Arabic Text", Libya, Pp.1-7, 2008.

[24] H. Harmanani, W. Keirous and S. Raheel, The International of Arab Jornal of information Technology, Vol.3,No.3, A Rule-Based Extensible Stemmer For Information Retrieval Application to Arabic, Pp.265-271, 2006.

[25] F.W. Lancaster, Online Information System, Encyclopedia of Library and Information Science, Vol. 20, Marcel Dekker, New York, 1977.

[26] Girma Berhe, Stemming Algorithm Development for Tigrigna Language Text Document, MSc Thesis, Addis Ababa University, 2001.

[27] Birungi, P., Improved Strategies for Employment and Human Resources Utilization in the Information and Documentation Sector, Strategies for Human Resources Development for Information Management in Africa, Ed. 49-57, Addis Ababa: UNECA, PADIS, 1995.

[28] Nega Alemayehu and Petter Willett, The Effectiveness of Stemming for Information Retrieval in Amharic for Information Retrieval: Electronic Library and Information Systems, Vol. 37, Num. 4, PP. 254-259, 2003.

[29] Lemma Lessa, Development of Stemming Algorithm to Wolytta Text, M.sc Thesis, Addis Ababa University, 2000.

[30] Wessel Kraajj, Viewing Stemming as Recall enhancement, In Hans-Peter Frei, Donna Harman, Peter Schauble and Ross Wilkinson (editors), Proceedings of the 19th Annual International ACM-SIGIR Conference on Re-search and Development in Information Retrieval, Pages 40-48, Zurich, Switzerland, 18-22 August 1996, ACM.

[32] Amit Singhal, Modern Information Retrieval: A Brief Overview, 2001.

[33] F.W. Lancaster, Online information system, encyclopedia of library and information science, Vol.20, Marcel Dekker, New York, 1977.

[34] Baeza-Yeates and Riberio-Neto B., Modern Information Retrieval, England, Addison Wesley Longman Limited, 1999.

[35] http://www.omniglot.com/writing/Silt'e.htm

[36] Ricardo Bauza-yates and Bertnier Ribeiro-Neto, 1999.

[37] Jelitan A.Hugh,E.Williams & S.M.M Tahaghoghi, Stemming Indonissian language, 28th Australian Computer Science Conference (ACSC), Conferences in Research and Practice in Information Technology, Vol.38.pp.1-8, 2005.

[38] Alemayehu, N and WillettP., Stemming of Amharic words for Amharic words for information retrieval, Literary and Linguistic computing, Vol.17, No.1, pp.1-17, 2002.

[39] Basen O.Alijla, Stemming in Natural Language, Faculty seminar, Departement of Information Technology System , Faculity of Information Technology, The Islamic Unversity of Gaza, 2009.

[40] Girma Berhe, Stemming Algorithm Development for Tigrigna Language Text Document, MSc Thesis, Addis Abeba University, 2001.

[41] Birungi P., Improved Strategies for Employment and Human Ressources Utilization in the Information and Documetation Sector, Strategies for Human Resources Development for information management in Africa, Ed. 49-57, Addis Abeba:UNECA, PADIS, 1995.

[42] Nega Alemayehu and Petter Willett, The Effectiveness of Stemming for Information Retrieval in Amharic for Information Retrieval, Electronic Library and Information Systems, Vol.37, num.4, pp.254-259, 2003.

[43] Lemma Lessa, Development of Stemming Algorithm to Wolytta Text, M.sc Thesis, Addis Abeba Univerity, 2003.

[44] Abebe Belay, Designing A Stemmer For Ge'ez Text Using Rule Based Approach, M.sc Thesis, Addis Abeba Unversity, 2010.

[45] Salton G. & McGill N.J, Introduction to Modern Information Retrieval, New York: McGraw-Hill, 1983.

[46] Hetzron R., “The classification of Ethiopian Semitic languages”, University of California, California, 1969.

[47] Hiava M.K, Cross language Retrieval–English/Russian/French at URL http://www.accessinn.com/aaai.htm, 1997.

[48] ELSE II project, Report on the feasibility of adding multilingual functionality to the search and browsing facilities of the ELSE system at URL http://nile.dmu.ac.uk/elise/el2_dels/d427d.htm, 1999.

[49] Peters Carol, “Multilingual Information Retrieval”, Presented at Cross Language Evaluation Forum, 2000.

[50] Kaplan, a Stemming Algorithm for Latin Text Databases, 2000

[51] Eiman Tamah Al-Shammari, Kuwait University, George Mason University, ISBN: 978-972-8924-63-8 © 2008 IADIS.

[52] Gutt E.H.M. & Hussein Mohammed, Silt'e–Amharic–English dictionary (with a concise grammar by E-A Gutt), Addis Ababa: Addis Ababa University Press, 1997.

[53] Gutt E.-A, “Concise grammar of Silt'e", in: Gutt, E.H.M. 1997, pp. 895–960, 1997.

[54] Dillmann, the Ethiopic language book, vo.2, n.3, p.107-112, 1958.

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

[56] Alemayehu N., Willett P., Stemming of Amharic words for Information Retrieval Literary and Linguistic Computing 17(1), 1-17, 2002.

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

[58] Argaw A.A., Asker L., An Amharic stemmer: Reducing words to their citation forms, In proceedings of the 45th annual meeting of the association for computational Linguistics, PP. 104-

110, ACI, Prague, Czech Republic Workshop on Computational Approaches to Semitic Languages, 2007 b.

[58] Dawson J., Suffix removal for word conflation, In Bulletin of the Association for Literary and Linguistics computing, Vol. 2(3), pp. 33-46, 1974.

[59] 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.

[60] Frakes W., R. Baeza-Yates, Information Retrieval: Data Structures and Algorithms Englewood Cliffs, NJ: Prentice-hall, 1992.

[70] Girma Berhe, "A stemming algorithm development for Tigrigna lanauge text documents", Master Thesis, Faculty of Informatics, Department of Information Science, 2001.

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

[72] K. Darwish and D. Orad, CLIR experiments at Maryland for TREC 2002: Evidence combination for Arabic-English retrieval, In proceeding of TREC 2002, Gaithersburg, Maryland, 2002.

[73] Khoja S. and Garside R., Stemming Arabic text, Computing department, Lancaster University, Lancaster, 1999.

[74] 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.

[75] L. S. Larkey and M. E. Connell, Arabic information retrieval at UMass, In proceedings of TREC 2001, Gaithersburg: NIST, 2001.

[76] L. S. Larkey Ballesteros and M. E. Connell, Improving stemming for Arabic information retrieval: light stemming and co-occurrence analysis, In SIGIR 2002, Tampere Finland: ACM, 2002, 2002.

[77] Lovins J., Development of a Stemming Algorithm, Mechanical translation and Computational linguistics, 11 pp. 22-31, 1968.

[78] Mayfield J. and McNamee P, 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, 2003.

[79] Paice C., Another stemmer, In proc. Of SIGIR Forum, Vol. 24(3), pp. 56-61, 1990.

[80] Popovie M & Willett P., Processing of documents and queries in a Slovene language free text retrieval system, Literature and linguistics computing, 5, 182-190, 1990.

[81] Porter M, An algorithm for suffix stripping Program, 143, Pp. 130-137, 1980.

[82] Savoy J., Stemming of French words based on grammatical categories journal of the American society for information science, Vol. 44, No 1, pp. 1-10, 1993.

[83] Xu J., Croft B., Corpus Base stemming using concurrence of word variants, In ACM transactions on information systems, Vol. 16, No 1, Pp. 61-81, 1998.

[84] Melucci M., Introduction to Modern Information Retrieval, New York, 2003.

[85] Massimo Melucci and Nicola Orio, the Measurement of Term Importance in Automatic Indexing 2003.

[86] Alemayehu, Nega and Willet P., Stemming of Amharic words for Information Retrieval, In

Literary and Linguistic Computing, 17 (1): 1-17., 2002.

[87] Paice D.C., Methods of Evaluation of stemming Algorithm Based on Error Counting, Journal of the American Society for Information System, 47(8): 632-649, 1996.

[88] Hmeidi I., Kanaan G. and Evens M., Desighn and Implimentation of Authomatic Indexing for Information retreaval with Arabic Documents, Jornal of the American society for information Systems, 48(10):867-881, 1997.

APENDECES

APPENDIX I: List of stop words compiled for the stemmer

አብተቴ እብታይ እብተቴ አብታይ አብተቴ እሊ አሊ እነይ አነይ

እቲ አቲ እቢ ዋ አቢ ሱር አድ አድአድ እነይ በልዳሌ አልቀሬ

ደር ኤት ፈሬ ፎኖ ግነ ቀደ ስር እንኮ ኡንኮ

ማ ምን አይኔ አይነኮ ላይሽ አይታይ ወይ ታሌ ናር

ናርት ናሩ ብቾ ለገነ ገነ ሁለምክ ለሳድባድከ ለሰባድሽ

ለሰባድኑም ለሰገግሽ ለሰገግኑም ለሰገግክ ለሰገጋ ሀዳድኑም

ድባየ አብሌ ናርት ናር ናሩ ቱፍትሉፍት ተሬር ታቼን

ቀለ በለ ቂጦ ለደር አበይ ጉት ኤጋህ ኤጋሽ ኤጋሙ

ታዮኑ ገናገናይ ዮናነይ ቢትላይም አሎነ ሂነሚ በሁነትነሙ

ሁኖተኒመዋ አታይ ሀነይ ሁኖ ገና ዮልስ ሉላሉሌ

በሉሌ አለቢ ኢንኩምንገ ቅጨ ብቾ ገገ ኡስጥ ግዝ

ሆነምታሌ በሁኖት ሉላሉሌ በልዳሌ ወክት ደር አይነኮ

ሉሌ አቢሌ ተደር በውኖትምክ የግግ ገገኑ ቲታሚ ኦናኔ

ገጌ ሀነግነ እነይ ሱርዋ ገናሚ ሂንኩምንገ ኡሀ

ያሽ ዬዬ በዬ ዋ ለገነገናም ለሂው በሰቼ አተም ግዝቸ

ሬራቀደን ሬራ ቀደን አናግና ኢንኪምንገ ሉሉሌ ዮለ

ሱር ሂንኩምንገ ለሀ ለሂ አዪ እታይ እተቴ አታይ አተቴ

ትዮኑ ዪታይ ዪተቴ ያታይ ያተቴ

APPENDIX II: List of Silt’e prefixes compiled for the stemmer

ኢለው እለው በል ኢለ በሰ አይ

እተ እት እል እለ አል አት አይ

አት ተይ በ ላ ቃ ጫ አ ሰ

ለ የ ይ ት ከ ቲ ተ ሻ ና ያ እ

APPENDIX III: List of Silt’e suffixes compiled for the stemmer

ኩሞሙ ቢያኔ አታም አተኘ አሙ ሼታ

ኩሙ ኒሙ ሼሽ ታት ኤን አኘ ተኘ ንቾ

እሎ ዬን ኔት ታም ተኛ ኩ ኢ ይ ቴ ኤ ነ ሽ

ኤ ኔ ከ ሽ ሽ ቸ ይ ኮ ኩ ሁ ው ነ ን ሽ ሺ

ኤ አ ተ ት ኡ እ እ አ ኦ ው ም እ ን ኤ ኢ

ቴ ቸ ሁ ው ሀ ሽ ነ ት ሙ ሁ ው ከ ሀ

አ ሽ ሺ ኤ አ ት ነ ሙ ኡ እ ኮ

ካ ን ም መ እ ይ ሚ ዋ ሶ ኒ ኝ

ሎ ያ ክ ኛ ቶ ቻ

APPENDIX IV: Silt'e alphabet

ሀ	ha	ሁ	hu	ሂ	hii	ሃ	haa	ሄ	he	ህ	hi	ሆ	ho
ለ	la	ሉ	lu	ሊ	lii	ላ	laa	ሌ	le	ል	li	ሎ	lo
መ	ma	ሙ	mu	ሚ	mii	ማ	maa	ሜ	me	ም	mi	ሞ	mo
ረ	ra	ሩ	ru	ሪ	rii	ራ	raa	ሬ	re	ር	ri	ሮ	ro
ሰ	sa	ሱ	su	ሲ	sii	ሳ	saa	ሴ	se	ስ	si	ሶ	so
ሸ	ša	ሹ	šu	ሺ	šii	ሻ	šaa	ሼ	še	ሽ	ši	ሾ	šo
ቀ	qa	ቁ	qu	ቂ	qii	ቃ	qaa	ቄ	qe	ቅ	qi	ቆ	qo
በ	ba	ቡ	bu	ቢ	bii	ባ	baa	ቤ	be	ብ	bi	ቦ	bo
ተ	ta	ቱ	tu	ቲ	tii	ታ	taa	ቴ	te	ት	ti	ቶ	to
ቸ	ca	ቹ	cu	ቺ	cii	ቻ	caa	ቼ	ce	ች	ci	ቾ	co
ነ	na	ኑ	nu	ኒ	nii	ና	naa	ኔ	ne	ን	ni	ኖ	no
ኘ	ña	ኙ	ñu	ኚ	ñii	ኛ	ñaa	ኜ	ñe	ኝ	ñi	ኞ	ño
አ	a	ኡ	u	ኢ	ii	ኣ	aa	ኤ	ee	እ	i	ኦ	o
ከ	ka	ኩ	ku	ኪ	kii	ካ	kaa	ኬ	ke	ክ	ki	ኮ	ko
ወ	wa	ዉ	wu	ዊ	wii	ዋ	waa	ዌ	we	ው	wi	ዎ	wo
ዘ	za	ዙ	zu	ዚ	zii	ዛ	zaa	ዜ	ze	ዝ	zi	ዞ	zo
ዠ	ža	ዡ	žu	ዢ	žii	ዣ	žaa	ዤ	že	ዥ	ži	ዦ	žo
የ	ya	ዩ	yu	ዪ	yii	ያ	yaa	ዬ	ye	ይ	yi	ዮ	yo
ደ	da	ዱ	du	ዲ	dii	ዳ	daa	ዴ	de	ድ	di	ዶ	do
ጀ	ja	ጁ	ju	ጂ	jii	ጃ	jaa	ጄ	je	ጅ	ji	ጆ	jo
ገ	ga	ጉ	gu	ጊ	gii	ጋ	gaa	ጌ	ge	ግ	gi	ጎ	go
ጠ	ṭa	ጡ	ṭu	ጢ	ṭii	ጣ	ṭaa	ጤ	ṭe	ጥ	ṭi	ጦ	ṭo
ጨ	ça	ጩ	çu	ጪ	çii	ጫ	çaa	ጬ	çe	ጭ	çi	ጮ	ço
ጰ	Pa	ጱ	Pu	ጲ	Pii	ጳ	Paa	ጴ	Pe	ጵ	Pi	ጶ	Po
ፈ	fa	ፉ	fu	ፊ	fii	ፋ	faa	ፌ	fe	ፍ	fi	ፎ	fo
ፐ	pa	ፑ	pu	ፒ	pii	ፓ	paa	ፔ	pe	ፕ	pi	ፖ	po

APPENDIX V: Ge'ez (Ethiopic) alphabet

ሀ	he	ሁ	hu	ሂ	hi	ሃ	ha	ሄ	hE	ህ	h	ሆ	ho
ለ	le	ሉ	lu	ሊ	li	ላ	la	ሌ	lE	ል	l	ሎ	lo
ሐ	he	ሑ	Hu	ሒ	Hi	ሓ	Ha	ሔ	HE	ሕ	H	ሖ	Ho
መ	me	ሙ	mu	ሚ	mi	ማ	ma	ሜ	mE	ም	m	ሞ	mo
ሠ	se	ሡ	`su	ሢ	`si	ሣ	`sa	ሤ	`sE	ሥ	`s	ሦ	`so
ረ	re	ሩ	ru	ሪ	ri	ራ	ra	ሬ	rE	ር	r	ሮ	ro
ሰ	se	ሱ	su	ሲ	si	ሳ	sa	ሴ	sE	ስ	s	ሶ	so
ሸ	xe	ሹ	xu	ሺ	xi	ሻ	xa	ሼ	xE	ሽ	x	ሾ	xo
ቀ	qe	ቁ	qu	ቂ	qi	ቃ	qa	ቄ	qE	ቅ	q	ቆ	qo
በ	be	ቡ	bu	ቢ	bi	ባ	ba	ቤ	bE	ብ	b	ቦ	bo
ቨ	ve	ቩ	vu	ቪ	vi	ቫ	va	ቬ	vE	ቭ	v	ቮ	vo
ተ	te	ቱ	tu	ቲ	ti	ታ	ta	ቴ	tE	ት	t	ቶ	to
ቸ	ce	ቹ	cu	ቺ	ci	ቻ	ca	ቼ	cE	ች	c	ቾ	co
ኀ	`he	ኁ	`hu	ኂ	`hi	ኃ	`ha	ኄ	`hE	ኅ	`h	ኆ	`ho
ነ	ne	ኑ	nu	ኒ	ni	ና	na	ኔ	nE	ን	n	ኖ	no
ኘ	Ne	ኙ	Nu	ኚ	Ni	ኛ	Na	ኜ	NE	ኝ	N	ኞ	No
አ	a	ኡ	u	ኢ	i	ኣ	a	ኤ	E	እ	I	ኦ	o
ከ	ke	ኩ	ku	ኪ	ki	ካ	ka	ኬ	kE	ክ	k	ኮ	ko
ኸ	`ke	ኹ	`ku	ኺ	`ki	ኻ	`ka	ኼ	`kE	ኽ	`k	ኾ	`ko
ወ	we	ዉ	wu	ዊ	wi	ዋ	wa	ዌ	wE	ው	w	ዎ	wo
ዐ	`e	ዑ	`u	ዒ	`i	ዓ	`a	ዔ	`E	ዕ	`I	ዖ	`o
ዘ	ze	ዙ	zu	ዚ	zi	ዛ	za	ዜ	zE	ዝ	z	ዞ	zo
ዠ	Ze	ዡ	Zu	ዢ	Zi	ዣ	Za	ዤ	ZE	ዥ	Z	ዦ	Zo
የ	ye	ዩ	yu	ዪ	yi	ያ	ya	ዬ	yE	ይ	y	ዮ	yo
ደ	de	ዱ	du	ዲ	di	ዳ	da	ዴ	dE	ድ	d	ዶ	do
ጀ	je	ጁ	ju	ጂ	ji	ጃ	ja	ጄ	jE	ጅ	j	ጆ	jo
ገ	ge	ጉ	gu	ጊ	gi	ጋ	ga	ጌ	gE	ግ	g	ጎ	go
ጠ	Te	ጡ	Tu	ጢ	Ti	ጣ	Ta	ጤ	TE	ጥ	T	ጦ	To
ጨ	Ce	ጩ	Cu	ጪ	Ci	ጫ	Ca	ጬ	CE	ጭ	C	ጮ	Co
ጰ	Pe	ጱ	Pu	ጲ	Pi	ጳ	Pa	ጴ	PE	ጵ	P	ጶ	Po
ጸ	Se	ጹ	Su	ጺ	Si	ጻ	Sa	ጼ	SE	ጽ	S	ጾ	So
ፀ	`Se	ፁ		ፂ	`Si	ፃ	`Sa	ፄ	`SE	ፅ	`S	ፆ	`So
ፈ	fe		`S	ፉ	fi	ፋ	fa	ፌ	fE	ፍ	f	ፎ	fo
ፐ	pe	u		ፒ	pi	ፓ	pa	ፔ	pE	ፕ	p	ፖ	po
		ፉ	fu										
		ፑ	pu										

ሏ	lWa	ጕ	gWu	።
ሗ	HWa	ቍ	qWi	።
ሟ	mWa	ኍ	hWi	፣
ሧ	`sWa	ኩ	kWi	፤
ሯ	rWa	ጕ	gWi	፥
ሷ	sWa	ቋ	qWa	፦
ሿ	xWa	ኋ	hWa	፧
ቈ	qWe	ኳ	kWa	:
ቧ	bWa	ጓ	gWa	
ዷ	dWa	ቈ	qWE	
ጇ	jWa	ኈ	hWE	
ጐ	gWe	ኰ	kWE	
ጧ	TWa	ጔ	gWE	
ቋ	CWa	ኧ	ea	
ቿ	PWa			
ፏ	SWa			
ፉ	fWa			
ፐ	pWa			
ቍ	qWu			
ኍ	hWu			
ኩ	kWu			

Figure 1. The basic Feedel (Ethiopic base characters and their six forms)

APPENDIX VI: Comparison of Silt'e words before stem and after stem

Silt'e words before stem

ቀደምሼታ'ሽ ሸ'ኔ ሺ.ፐት ቀደምሼታ'ሽ አያሜ አያሜ አዮለ'ሽ በቡርደ'ሽ ያለትወደ ያለትኮ ማ ኤወደ ለጦራሚ ኡርበ ለሳንኩራይ ኢለውነቅ የሀደኡመት ኪሼከ በለ.ለ.ሌ ስንፌ ዬውዲበያን የቀውልን ሱብሱብ ለፋፍነት ለገረድነት በሚ'ሽ ሚ'ሽ ገፈራኔ በሚ'ሽሚ'ሽ መዬ በሬደ የሙሃበ አይጦፋን ሼጣኔ ሰመይ ወዘና ወልዴ አያም አተረ በሀቅል መርከተኛ አልበለይ ኢሼማን በጪልመ የጠለሀ አስለመጠ መረጩከ ኪዝበኛን አራሺን የቀበጠ አዋለኩ ወከቡ ተቀቡ ጩጠ ይልብ አዋልከ ሀድ ሰብ በቀውል አናጊን በኩትብ ለያቲለልፍ ኢከሸነይ ለክተ ፈያክ አሻኔ ለሃ የትሬ ቂጨ ለገና ሰብ ኢትሪናኮዋ ኢቻለነኮ ያሸቢያነን ዋልከ አይነት ስነ ዱትብ ጡሪ በለይ አዋለከ በቀውል አናፈን በኩትብ ኢቀርባን ግዝ አዋልከ ሊግኖት ለሰብ ወልድ ቢሃል ቀይማን ዋ ቢያቀር በይማን ፍደልቸ ኡስቤከ አናፈና ለክታከ ለቲላል ፎት በኩትብ ኢትነበለን የቀውልን ሱብሱብት ኩትብ በስንፌ አናጊን በቡትን ሊቀርብ ያቀትለን ኩትብ ወገሺን ተገደገደ ቲለልፋን አለመ አበ አዬነ'ሽ ኪቲሰል ወራቤ በሂዶት ደር ታለት ቴድበያቲ መኪና ታብለሻኔ ጉራይ ኢንግ'ሽ የትሜለ የተረሻት መሀመድ ሞተ ሚ'ሽተከ ከዲጀም ሞት ቢለ መሰው ሆ'ሽት ሰብቻ ለሚነ ሞቱ ለሚነ ሊሞቱ አቀተሉ ኢለንሱል ሊነቀ ያቀትለን ኩትቢ ለቀረበይ ሱል ኪምባዩ ለዩባናኮ ኢለጦፍ ሎናምከ መሀመድ መኪነ ታግለበጠቢያኔ በምዶትከ ሚ'ሽትክ ከዲፋ የሚ'ሽ ሙዶት ሰማታኔ ቱሃ ተለዬው ኢለውነብር ባታኔ ኢቀትለን ደዌ ሰቼት ሞት ኢታይ ኩትብ መንቂዋ ውጣት ሌንዘከ ለቀሪሎይ ያጦፋን ቢቢ መሳወ መሀመድ የሞተቢመንቂከ የከሚናን አምሰ የሚ'ሽትክ መውት መንቂከ ደዊዋ የመሀመድ ተመውት የስልጤ አፈርከ ኢቤዣን ለላዶ ዘማን በቴለቀ አ'ሸር በጤማይ ባድ አርከ በታለቀ በደውሰይ በድ ኮሌጅ መኒ ባሉነ ወራቤ ቢሌድ አጋኔ አይቺሌን ግዝ ኢነካሌ አማድ ኡመተው በድሁን አለሃን አሀምድ ለታንጩ ለኖታም ጫፍ ሲር ቲያበሊ አለ ሚነ ኢላቴር ሰበው በርደባሊ ሃኪመ ጋር ለርከቦት ራቂ ቲትጉላሊ ተውኖት ፈመረ ቦራቤ ሆስፒታሊ አብሺሪስ የ'ኘቴ ወራቤ ከተማ ሰሙይ በቄበበይ ኢዝነ'ሽ ተሳሴማ ሙሶ ቀ'ኜቢሻን ሱማ'ሽ ሉልም ጫም የሲልጤን ወልድ ሆናን ወራቤን ያለንገፍ አደም ጊዝ ታያቀር አለሃው ዋበ'ሽ አሮት ማልተ ታይቢል በጃደ ወልደ'ሽ፣ ቶቢያ ባዴያ ጌሰም ቲሌቄ'ሽ ኪመ አበቃሎ የባነ አፈረሻይ በሂም ለ.ለ.ሌ ኤንዘን ገዎሻይ ከለፌ አለፈዱይ በለንኒ እመሻይ ያፍሪቀቴን ወሪ ቶቢ በድናይ ለዚጠ'ኜ ወሪ የለበስኩይ ጂስማ'ሽ ሆ'ሽተ ዛን ዱነከይ የጠወረኝ ጪነ'ሽ ኦሮሮ ቲትቢይኝ የራመ'ኒ አንጀ'ሽ አለሃን ኢጦቅሳያው ለያቴሪኝ ቡቻ'ሽ ዱማነ'ሽ ሹሩበ አቅናኔ ታቴሻን

ቢሴቹ በቅባት ፈያክ ሚሰሱሻን የሰመይ አመረ ሙግ በለ ያንገርሻን ኢታይ ታክማመቶት ላሸው ሚነ ያኚንሸን ሲልጢነታሽ መጢክ ኢዝኪክ ኢከታሻን አዝጋግ በሎት በቡርደነ ሀነጊነ ባቢሌ ይትረከቦን ሀለትቸ ዬንዜ ሀደ ግዝ ሆኖኔ ሰብ ሀነጊነ ሩህ ያለይ ገነ ጊዝ ሀነጊነ ያአመኜ ሰብ ሩክቦ ያሺያነን ጊዝ ዩልቀተኜ አዝጋግቸ የደች ቃምቸ ለባይትክ ቆቶ ወንደበን ሚዴ ገነገናም ዬንዜ በሂምቸዋ አበቃላሎነ ቡንዱላሌ ዬንዜን ሰበኜ አዝጋግችንገ ያቀኑይሙ አዝጋግችነ ቢቾ ተዮን ያመኜ አዝጋግችነ ለባይትክ የአህላቅ ሀለትችነ ሲያሰኜዋ ኢኮኖመኜ ሀለትቸዋ የአደኜ አዝጋግችነ ለባይትክ ሰብአኜሎ አቆምስቸ አርት ዲን ሰብጊቾ ሳይንስኜ ቴክኖሎጂኜዋ ላቁምሶት ዬድግሎን ጊዝቸ ዬንዛን ያዝጋግ ሳይንስ ባድለኜ ሳይክለ በፈትለኜ አሸር መቃም ያደ አሸረኮ ለደረሳሶ ቲቀርብ ለደር የጨቀሙይሙይ ጊዝችነ ባደኜ ቡንቁፌ ዬንዛነኮ አቻበሩያን ለሆነም ኤት አሸረይ ሉላሌ አሸርችነ አቻበራኔ ኤንዛን ለባይትክ የጅስመ ሳይንስ የሃያት ሳይንስ ያድኖት ንባረት ሳይንስ ያመኜ ሳይንስ አፊየ ሂርሰት የንባረት ብላት አሸርዋ ዬንጅ ብልችነ አቻበራኔ ዬንዛን ያዝጋግ ሳይንስ በጢሸት ያቀራነይዋ ጉተኜ ሂንገፍከ በሰብያዮ አዝጋግ ሃለትቸ ደር ይትሰበታነይ ሰበዬይ ቲሌቅ ቲዬድ ተቡርደክ ነቀላኔ ቲፌት ዬዳነይ አዝጋገክ ፊልፊሎት ይጀሚራን ሂታይ ሊቆት አዝጋገክ የቻሎት ፊልፊሎት ሃለት ሄደ ሄዳኔ ቱህ አቢሌ ታለይ ፈቲይ አዱኒያ ተራክቦትክ ለሰቢዬይ መሊቅዋ ተጀረቦት በሉላሌ ኡንገ ይድገለያን ሂ ሰቢዬይ ተቡርደክ ጀመራኔ ለቻሎትዋ ገገክ ለትጀረቦት ያሼያነይ ጅህድ ሰቢዬይ በቡርደክ ለቻሎት ያለይ ቆመፌቸ አቁሚሮት ደበላኜ ገገክ ተፈቲይ አመኜ አዝጋገክ ለቻብሮት ያድገለያን ሰቢያዮ ቢታይ ሃለት በሰብቸ ጉት ያለይ ሩክቦዋ ሃድነት ለትጀረቦት ተድግሎትምክ ደበላኔ በባድቴ ያሉይ ሉላሌ አደቸዋ ሰብችነ የንበሮት ሃለትቸ ለትጀረቦትዋ አደኒሙዋ የንበሮት ሃለትኒሙ ላሂብዶት መሰ ዮናን ሂነሚ ያትሬጊጣን በድባያም የአዝጋግ ሳይንስ አሸር በሰወዶም የሁን ባመኜ አዝጋግነ ያክማምቶትዋ የትቃቂሮት ወሻይብነት ይነብረናነኮ ያሻን ያዝጋግ ሳይንስ ሰብያዮነ ገገኒሙ ለትጀረቦት ይድገላን ኢልምዋ በሰበ ጉት ፊየ ዮነ ርክቦ ያሶነኮ ይግዘይማን ትዮን ፊየ አሸላቅም ሊነብረይሙ መሳን አሸረይ የሰወዶም የሁን ያመኜ ፊየ አህላቅ ባትሪክቦት ሰብያዮነ ሳይንሰኜ የሱል ብላትቸዋ ዬምርዋ የሃያት የቻሎት ኢልምቸ ለርክቦት አሰነት በውኖት ቆማሬ ዮነ ፈይሰለ ባቦት

ያውጀም የሁን የከሸፈ ወቅተኒሙ በሱተ ይድግሎነኮ መሰ በውኖት ፈይደ ዮበይማን አሸረይሚ ሰቢያዮነ ለባድኒሙ ይድጋሎን ጠረኜ ሰብ ያሺይማን

ያዝጋግ ሳይንስ ያሸር ንድፌ ጎልጌ ጃንጌ ቲትሲናድ ባጂሲይ ያሸር ፎሊሲ ለቶጲ ያሸር ጋርቸ ቢቤሹን ሃለትዋ ለቀደ በትረሼይ ሁንዱለሌ ዞፎፎ አሰነትን የንድፌ አሸረይ በቂጦ ኡምር ይትረከቦን ሰቢያዮዋ በቂጦ ጎልጌ ላሉ ደረሳሶነ ኡስቤ ባሶትን ኢትቂራነይ የቻሎት መቃምዋ ኢንገፍይ ጎልጊቸ ቢከተይማን የቻሎት መቃም ሃለትን ተስናጃን ሂርሰት ቴክኖሎጂዋ ኤዲሰ በትረሺይ ሁንዱለሌ ዞፎፎ አሰነት ቢቤጣንዋ ቢድግሎን ቂጨ በፊቲክ ሃደኘ ይቻበሮነኮ ባሸሪይ ውስጥ አገባይማን ንድፌ አሸሪ ለደረሳሶይ ለአቅልኢሙ ቢያብያይናን ሃለት ያለ ግዝ ባውጦት ለደረሳሶይ ቢትፋሃመኢማን ሃለት አጣሚያን

ሀጂሰይ ያሸር ከሪኩለም ለቶጲየ ያሸር ጋርቸ ቲትሲናድ ሰአብት ሲጀ ወቅቲችነ ባድ ሳምት ኡስጥ ለአዝጋግ ሳይንስ አሸር ሳደያን ያድ ያሸር ዘማን ሳምቲቸ ሲጀ ወቅትቸ ወሰዳኔ ለትሲናጂ ኤት አሸርጌታቶይ ፈት ወቅት ይነብረይማነኮ ባይትም ዲባያዮ ሳምትቸ አትረከቢኢይማን ቢጹታሚ ወቅት አሸርጌታቶይ ፈት ኩፍተ ሲጀ ወቅትቸ ሊረክበንኤት ሚካት ያለቢይሙዋ ሉሌ ኡግሹር ያትኬሼይማን ደረሳሶነ ሊገዞት ያቀትሎነኮ መሰ ዮነይማን ኩፍት ወቅት ኒበሮትከ ደረሳሶይ ሊገዞትዋ ገነ ዲባያ የአሸረ ቢል ላሶት መሰ ተሁኖትምከ ደበላኔ የአሸረይ አይዶ በሱት በድጋለሎት የትጎበለይ ቀስድነ በሱተ ለከሚሎት ይድግላን

ለሆነም ኤት አሸርጌታቶይ ኩፍተይ ወቅት አሸረይ ቀደሞኔም ቢፊዱ በማን ማነምግዝ ታቲልፎት ለደረስቻይ ኡግሣር ቢያኛን ሃለት ተድጋለሎት ይነብረይማን ሃጂሲይ ፎሊሴ የአሸረይ ሃለት ቲዬውድ የቅሮት አቅሮት ሃለቲይ ዩለጊነ ዞፎፎነ ባዴኘ ባቻበረ ሃለት ኢንዞት አለቢይን ይላን ሂነሚ ለከውኖት የቀውል የክተቦትዋ የከውን ሃለት ተድጋለሎት መጥን ለሆነም ኤት ሂ የአሸር ንድፌ አሸርጌታቾ ዩለጊነ ዞፎፎነ በጎልጌ አሸር ቲያቀሩ ይትዲገለሎነኮ ኡስቤ ያሻን ቲዮን ኡሀነም የጎልጌ ቢል ባፍዋ በኩትብ ጃድ የጋረ ቢል ባሶት ዋ ያቢሌ የከውነ ብል ባሶትዋ ዞፎፎ ባሶት በሰመጢከ ደረሳይ ቢል ያሻነኮ ባኮትዋ ባዙፎት ገነገናም ከውኖትን

ቢታይ ንድፌ አሸር በሱራይ በሰወዶ መትፈጂከ ኤት ላዙፎት ይድግሎን ቀውልቸግ ፈሀሚቸ ያሉ ቲዮን አሁንም አሸርጌታቶነ ሙላይ ያሸረይ ቀስድ በሱተ የትከመለዋ ያልቲከመለኮ ላጢሮት ይቤሹይማን ቢያግህሮን ወቅት ኢከታን የቻሎት መቃም ቂጨ" ይላነይ ኡስቤ ገነ ኤት ውሰዶት ኤለቢይ ለበሎት የከሲይዋ ተጀረቦት ያለቢነይ ሱተጄይ ቢልዋ በሱተይ መቃም ሀድ ደረሰ ተጀረቦት ያለቢይዋ ለኢታይ ላለቢይ ጎልጌ ቂጨ ዮነ ከሚሎት ያለቢይ ኢልምን ለበሎትን ቲታይ ቂጨ ከሎም የሁን ደረ አዘረክ ያሉ ደረሳሶነ አሸርጌታቶይ ልቤ በሎት አለቢይሙ በደርከ ያሉይ ያቆምሮነኮ አንሻሸጦት ያለቢነ ቲዮን በኮሎ አዘርከ ያሉይ ጋኔቶ ሚካተኢሙ ላሌኔ

ኡግጎር አሶት ያትኬሻን ሂነይ ቀስድ በሱተ ለከሚሎት አሸርጌታቶ ደረሳሶ ያሸርጋረይ አሚርቸዋ የአመኜ አዝጋግ አበሮስ ሁሉምከ በቂጠ ተቻበሮኔ ጃድ አሶት ያትኬሻን ሂነይ ያሸር ንድፌ ጎልጌ ጃንጎ በሉሌ የአዝጋግ ሳይንሰ ያሸር ንድፌ የትሴናዱይ ያጥቃቀሎይ የሙለይ በቶጵያ ያሸር ሚኒስቴር ያሸር ንድፌ አስናዶትዋ አቆምሮት ጎልጌ ቲዮን በፈረንጂይ ሂልቅ ያሰናዱይ ለደር የቀረቡይ ባይትቻነ አይነኮ በነቶ ለይቴንዚ ያግዘነኮ ባበይኖት ወባጃይ ጀምሮት ለሰአደዲይ ነታቶ ደረሳሶይ ሚነ ተረሶት ያለቢይዋ ተቀጠብቻነ ዬሰቦነኮ አሶት የሳንበ ነቶ በአየር ይትብራጫነይ፣ አይነኮ ይትቃጠብነያን? አየርንግ አግቦት አውጦትነ አቃኖት ኢላቀትሊ፣ ጊንመ የጋረነ አየር መሰከተከ በክፈቶት ጠሊል አሶት ያቀትሌን የደል ሙርጠጤ ሹርት ይሰራጫነይ ተዌሴ ሹማትን አይነክ ይትቅራቀርነያን በሸማኒ ዞፍ በሱተ እንደ ተራጦት አለቢነ በኒጥር ኢንጅ ሲንቅ ያቀረበ ሰብ ስንቀከ ብሎት ኤለቢና ሪጀ ይትሲራጫነይ በሪጀ ጨንጪ ቲነከሲን አይነኮ ይትቀራቀርነያን በሪጀ አጎበርዋ የቃነነ መይ ባፊሶትዋ በድሪቆት ኤድሰ በደምን ይትሲራጫነይ አይነኮ ይትቂራቀርነያን ደመነ ተገነሰብ ደመኒካኮት ኤለቢነ ተዚነዋ ሱጠ ዮኑ ሙትቸነ ደመ በነኩዋ በቂጠ ተድጋለሎት ቢደጎት የጅስመ ቆመፌቻች ሚነ ዮኑኮዋ ሱመኒሙ ጪቅሞት ደረሳሶቻይ የሰአደዲይ የጅስመ ቆመፌቻችነ ይጨቅሞኔ ያቴሮነኮ ይጠሩይማነኮቢለኒይሙ ያበይኖነኮ አሶት ደረሳሶይ በገገኒሙ ተሮሻት በነቀ ይሸሉይማን ዮኑ የኑርቡቃቆነ ይጠሮነኮ አሶት ለላሉሌ ዮኑ ለባይትከ ምስለ በትዲጋለሎት ሃቀኜ ሙትቸ ሻመ ካሌ የኮረንቴ ኑር ሙቃድ ይነዳን ኢንጩ አይርዋ ወሪ ባቲሮት ደረሳሶይ ለላሉሌ ኑርችን አቲሮት ሎነኮ ተሳሎት የነታቶይ ሂልቀኒሙ በድሎት ደረሳሶይ አይታይ ሱር በቡርደኒሙ ሁለጊነ ይትረከቦንወ በለ አይትረከቦን በብሎት ይለሎነኮ አሶት ደረሳሶቻይ ኤድሰነ ቲገናይ በጠቀስ ፈየኮ ለይችሎም ያቀትሎን ነቶይ ቀታይዋ ደዊ ዬለይ ሆኖተኮ ኢውዶት ለኢምኮ በቡርደኒሙ ተገናይ ቲያትዋሩይ በብኸር የልቲረከበይ በበሎት ለደረሳሶይ ለሰብ ወልድ ሲንቀ ላሊቆት ለቁሚሮትዋ ቁወ ላቦት ያድገለያነኮ አበይኖት ደረሳሶቻይ ይትረዘቆነይ ቶኑይ ሃዳላን ሴንቂቸ ኡስቀ ሱመኒሙ ይጠሮነኮ አሶት ዮዱያነይ ሲንቀ ሱመም ይጠሮነኮ አሶት

Silt’e words after stem (stop word exclu

ሸኜ ሺፐ ቀደ ያሜ ዮለ ቡርደ ማ ኤወደ ኡርበ ውነቅ ሀደ ኪሼ ለሉሌ ስንፌ ዬውዲበ ቀውል ሱብሱብ ሚሸ ገፈራ መዬ ሬደ ሙሃበ ጠፋ ሼጣ None ወዘና ወልዴ ያም ተረ ሀቅል መር ለይ ኢሼማ ጪል ጠለ ስለመጠ መረጩ ኪዝበ ራሺ ቀበጠ ዋለ ወከቡ ጩጠ ዋል ሀድ ሰብ ጊን ኩትብ ፈያ ሻኔ ኡሃ ትሬ ቂጨ ኢትሪና ኢቻለ ሸቢ ነት ስነ ዱትብ ጡሪ ፊን ኢቀርባ ግዝ ግኖ ወልድ ቢሃል ቀይማ ዋ ቢያቀር ፍደል ኡስቤ ፊና ኡክታ ፎት ኢትነበለ ቡት ቀርብ ቀትለ ወገ ሊመ አበ ዴነ ኪቲሰል ወራቤ ሂዶ ደር ታለ ቴድበያቲ መኪና ታብለሻ ኢንግ ሜለ መሀመድ ሞተ ዲጀ ሞት ቢሉ መሰ ሆሸ ሞቱ ቀተሉ ንሱል ነቀ ኩትቢ ሱል ኪምባዩ ጠፍ ለኖ መኪ ታግለበጠ ምቶ ዲፋ ሙቶ ሰማታ ቱሃ ውነብር ታኔ ኢቀትለ ደዊ ሰቼ መንቂ ውጣ ሌንዘ ቢቢ መሳወ ሞተቢመንቂ ሚና ምሰ መው ደዊ ስልጤ ፊር ኢቤጣ ዘማ ቴለቀ ሸር አድ ርከ ታለቀ በድ ኮሌጅ መኒ ሉነ ቢሌድ ጋኔ ቺሌ ኢነካሌ ማድ ድሁ ሀምድ ጫፍ ሲር በሊ አለ ሚነ ኢላቴር ሰበ ርደባሊ ሃኪ ጋር ራቂ ጉላሊ ፈመረ ቦራቤ ሆስፒታሊ ብሺሪስ ኘቴ ተማ ኢዝ ሙሶ ቀኜቢሻ ሱማ ኡል ጫም ሲልጤ ሆና ደም ጊዝ ታያቀር ዋበ ሮት ታይቢል ጃደ ቶቢ ዴያ ጌሰ ሌቂ ኪመ ቃሎ የወ ሂም ኡለሉሌ ኤንዘ ለፌ ፍሪቀ ወሪ ጂስማ ዛን ጠወረ ጪነ ኦሮሮ ቢይ ራመኒ ንጀ ኢጠቅሳ ቡቻ ዱማ ሹሩበ ቅና ታቱሻ ቢሴቼ ቅባ ሚሰሱሻ መረ ሙማ በለ ንገርሻ ታክማ ሸው ኚን ሲልጢነታ መጢ ኢክታሻ ዝጋግ ሎት ሀነጊ ቢሌ ሀለ ዬንዜ ሆኖ ሩህ ገነ መኜ ሩክቦ ሺያ ዩልቀተኜ ደች ምቸ ቆቶ ወንደበ ሚዴ ገነገና ሎነ ቡንዱላሌ ሰበኜ ዝጋግቸንገ ቢቾ ህላቅ ሲያሰኜ ኢኮኖመኜ ደኜ ሰብአኜ ቆምስ ርት ዲን ሰብጊቾ ሳይንስኜ ቴክኖሎጂኜ ቁም ዬድገ ዬንዛ ሳይንስ ድለኜ ሳይክለ ፈትለኜ መቃ ያደ ሸረ ቡንቁፌ ቻበሩ ኤት ኡላሉሌ ሸር ቻበራ ኤንዛ ጅስ ሃያ ድኖ ንባረ ፊየ ሂርሰ ብላ ዬንጅ ብል ጢሸ ጉተኜ ሂንገፎ ብያዮ ሃለ ሌቅ ዬድ ነቀላ ፌት ዝጋግ ፊልፊ ቆት ቻሎ ሂደ ሂዳ ቱህ ዱኒ መሊቅ ለላሉሌ ኡንገ ሂ ጀመራ ገገ ጅህድ ቆመፌ ቁሚሮ ደበላኜ ድገለ ሰቢያዮ ብቸ ጉት ሃድ ደበላ ባድ ሉይ ደቸ ንበሮ ደ ሂብዶ ሂነ ሬጊጣ ድባ ወዶ ሁን ክማ ቂሮ ወሻይብ ሻን ሰብያዮ ኢል በ ፊየ ርክቦ ሶነ ዮን ኸላቅ መሳ ሰወዶ ሪክቦ ሳይንሰኜ ዬምር ሰነ ውኖ ቆማሬ ፈይሰለ አቦንድፌ ሸር ወዶ መትፈጀ ኤት ዙፎ None ቀውልችግ ፈህ ያሉ ዮን ሁን ሸርጌታ ቀስድ ሱተ መለ ልቲከመለ ጢሮ

ቢያግሀሮ ወቅ ኢከታ ቻሎ መቃ ኡስቤ ገነ ውሰዶ ሲይ ቢል ሀድ ደረሰ ጎልጌ ቂጨ የወ ሚሎ ኢል ታይ ሎም ደረ ዘረ ደረሳ ልቤ ሎት ደር ሉይ ቆምሮ ንሻሸጦ ኮሎ ዘር ጋኔ ሌኔ ኡግሃር ሶት ኬሻ ሚር መኜ ዝጋግ ሮስ ሁሉ ቂጦ ጃድ ጃንጎ ሉሌ ሳይንሰ ሸር ቶጵ ሚኒስቴር ስናዶ ሂልቅ ቻነ ነኮ ነቶ ግዘ ጀምሮ ነታ ሚነ ዬሰቦ ሳንበ የር ርንገ ግቦ ውጦ ኖት ጊን ጋረ ክፉ ጠሊል ቀትሌ ደል ሙርጠጤ ሸር ሸማ ነክ ሸማ ዞፍ ንደ ኒጥር ኢንጅ ሲንቅ ቀረበ ሰብ ስንቀ ብሎ ኤለቢና ሪጀ ጨንጪ ነከሲ ጎበር ነነ መይ ፌሶ ድሪቆ ኤድሰ ደም ደመ ኤለቢ ሱጠ ሙት ነኩ ቂጠ ቢደጎ ጅስ ቆመፌቻኞ ወዋ ሱመ ጪቅሞ ቴሮ ገገ ነቀ ኑርቡቃቆ ኡላሉሌ ምስለ ዲጋለ ሃቀኜ ሻመ ካሌ ኮረ ኑር ሙቃድ ኢንጩ ርዋ ወሪ ሮት ሎነ ሂልቀ ድሎ ሱር ቡርደ ሁለጊ በለ ረከቦ ጠቀስ ፌየ ቀት ደዊ ኢውዶ ብኜ ወልድ ሲንቀ ሊቆ ቁወ ቦት ድገለ ሃዳላ ሴንቂ ኡስቀ

Appendex VII words with lower frequencies but are stop words

Word	frequency	Word	frequency	Word	frequency	Word	frequency
ገናኒሚ	2	ልቶን	3	ኢነ	2	ጊዝቸ	2
ጃንሦ	3	ቦንት	2	ኢንኩ	2	ሂንኩ	2
የሰሀደደይ	2	ያፍቱቱ	2	ኢነኮ	2	ዩሃን	2
ለገጊኒሞ	2	እቱታን	1	ባታይ	3	በገጊሽ	2
አደደሞ	2	እነይ	2	ገነግነ	2		
ሂተይ	2	እታይንገ	2	ሎኔት	2	ለሆነ	2
ጊንመ	2	ሊሊ	2	ሂንኩምከ	2	ሂንኩም	5
አደኒመ፡	2	ዮናነይ	2	ታት	2	ዩኑ	2
ሊኢታይ	2	ገገሽ	3	ለዜት	2	ዮናን	6
ቲቲይ	2	ቦነንገ	2	ሚነ	3	ለሆነምከ	2
ኡሃም	2	ወናኔ	2	ዩኑኮ	3	ያፍቱ	3
ተደር	2	ብቹ	2	ሂንኩሞ	3	ናር	2
ሬረ	3	የብቸ	1	ታይታይ	4	ተዮን	5
ሆኖት	2	በዮን	2	ዩነኮ	2	ሊታይ	3
በገጋመ፡	2	እሊ	2	ለሚን	2	በምን	5
በዮንም	2	የኛነይ	2	ቢቻ	1	ሎንት	4
ሂንካሌይ	2	ባሎትን	9	ሂ	3	ቢታይ	13
ገናኒሚ	2	አዲ	6	ሂነይ	10	ሁኖተኒመ፡	3
አነጊነ	7	የገናይ	6	ለደር	9	በገና	3
ዩልይ	2	ሂንኩምንገ	4	ታዮኑ	2	በሮሬ	3
ዮነይ	4	የሰአደደነ	2	ሊትኬተሎነይ	9	ገገኒመ፡	6
ለታት	8	ገገከ	5	ዩነይ	4	ቲኢኮሎ	3
አይኔ	2	ገነገናሚ	2	ኢታይንገ	2	ሁልም	6
እታይ	2	ተገናይ	10	ገናም	3	አዘር	14
የዜተ	2	ኡሁኑም	6	የገገከ	7	ሁሉም	3
ሀኔ	2	የናረይ	8	ለገናሚ	3	በሁኖተኒመ፡	3
ለኢኮ	6	ኢነይ	9	ገነገናምኒመ፡	1	ታቲ	3
ሆነም	4	ሂሊኮ	2	ገግ	4	በገግኒመ፡ዋ	3
ታሌ	4	ሎነም	9	አይነት	16	የገናም	3
አልዳሌ	3	ኢታይ	12	ግን	3	ቀደ	6
አዮናነይ	2	ጊነ	6	ኡስቤ	26	አቱም	3

አይታኒ 8	በገነ 7	ለሳድባድ 2	ዮነን 4
የገነ 2	ሂታሚ 4	ገና 2	በሁኖት 3
ዞፍ 9	ሱር 9	ያበደም 2	የሁን 5
በጉትኒመ 6	ጉተ 2	በአበድም 2	ሁሉምግነ 2
በባይት 2	ሩክቦ 22	ለምን 8	ቲኢ 5
ገናይ 9	ግዝቸ 11	ገነ 9	ላፍቱ 3
በአድ 3	ሁለም 5	ለሰሀአደዲ 2	ኮሎ 16
ሂታይ 5	ባፍቱ 2	ምን 12	ዮኑ 13
ለሰሀደዲ 3	ገነይንገ 2	አይነኮ 13	ኡሁንም 2
ቶህ 3	ታቲም 3	ጉት 16	ጉተኜ 2
የቶህ 3	ሁኖተከ 11	ለሰደዲ 2	ሀለት 14
ግነ 13	በልዳሌ 12	ሀደደ 4	በጉደር 2
ሬሬሰ 5	ቢኢታይ 3	በገግ 6	ለባይትምከ 7
አሶት 16	ኢንኩምንገ 7	የገግ 5	ያሌተ 2
ቢኢሸ 2	አድባድ 3	ለገናይ 6	ሁለምግነ 2
የሳድባድ 4	ለሆነምኮ 3	ምነ 13	በጉተኜ 2
ለሰደደኒመ 2	በውኖት 7	አይታይ 5	

Declaration

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

_____

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

_____