ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

SCHOOL OF INFORMATION SCIENCE

Linguistically Motivated Amharic IR (LM-IR)

A Thesis Submitted to the School of Graduate Studies of Addis Ababa University in Partial Fulfillment of the Requirements for the Degree of Master of Science in Information Science

By

Biruk Demelash

June 2013

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

SCHOOL OF INFORMATION SCIENCE

Linguistically Motivated Amharic IR (LM-IR)

A Thesis Submitted to the School of Graduate Studies of Addis Ababa University in Partial Fulfillment of the Requirements for the Degree of Master of Science in Information Science

By

Biruk Demelash

June 2013

ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
SCHOOL OF INFORMATION SCIENCE

Linguistically Motivated Amharic IR (LM-IR)

By

Biruk Demelash

Name and signature of Members of the Examining Board

	Name	Title	Signature	Date
1.	_____	Chairperson	_____	_____
2.	_____	Advisor(s),	_____	_____
3.	_____	Examiner,	_____	_____

Dedication

To my best friend Teke, thank you for all the commitment you have shown in my life!

Acknowledgement

I would like to acknowledge my Advisor Dr. Million Meshesha for his diligent guidance, even before this research. The best instructor I have ever seen. Let the Lord bonus Him, for what he has done without seeking reward. I also like to appreciate my junior brother Henok, who assisted me in preparing the corpus in the required format.

My wife also played a major role for the accomplishment of this thesis, thank you and I love you. The Baby (Rebecka), now, I will have more time to play with you.

Table of Contents

List of Tables

List of Figures

List of Algorithms

List of Acronyms

- IR: Information Retrieval
- NLP: Natural Language Processing
- LMI: Linguistically motivated Indexing
- NMI: Not Linguistically motivated Indexing
- ASCII: American Standard Code for Information Interchange
- UTF-8: Unicode Transformation Format
- LMIR: Linguistically Motivated Information Retrieval
- LA: Linguistic Analyzer
- LC: Linguistic corpus
- SAA: Semantic Analyzer Algorithm
- EGA: Enhanced Graphic Adaptor
- VGA: Video Graphic Adapter
- ISO: International Standardization Organization
- TSA: Tree Structure Analysis
- CV: Conceptual Vocabulary
- MSA: morph syntactic analysis
- NLII: Natural Language Inherent Indexing

Abstract

Information Retrieval (IR) is the very essential tool in every society for knowledge acquiring. The challenge of designing effective IR on Amharic is related to linguistic characterstics that are specific for the language.

Detail studies on the Amharic language indicate two core features. These features make difficult to apply IR models that are effective on English. The first is syllabic nature of the writing system the other is morphological nature of word formation. These characterstics cause too many morph variation and linguistic ambiguity. That is why applying already existing IR models cause document silence and noise during.

Adopted models of statistical preprocessing fail to give enough attention for the core characteristics of the language, in this research an attempt is made to develop a new Linguistic Analyzer (LA) for word preprocessor using morph syntactic analysis (MSA) to resolve challenges related with linguistic ambiguity and linguistic variation.

Morph variation has been a major challenge of Amharic IR system by causing document silence during retrieval. This problem has been resolved in this research by introducing incremental index file structure. Incremental indexing has a capability of storing linguistic inflections that are related with gender, number, tense, and other form. This indexing structure helps to keep precession while increasing the recall values of retrieval system.

A preprocessor LA is build using 74,000 words found in Amharic bible. After performing preprocessing on 5000 words using the newly designed LA, output found with better performance of 82%. On the same test the statistical preprocessor with stemming can deliver only a maximum of 30%. The LM-IR, that is built on top of LA have incremental indexing file structure that is capable of delivering average F-measure of 83%. It was possible to maintain recall of 88% while the precession is not below 76%

The comparison of LA and statistical word preprocessor shows a significant difference on effectiveness therefore LA approach benefits Amharic IR design. In addition the incremental indexing structure protect the semantic lose on index words that used to happen statistical index structures. Incremental indexing structure helps to increase recall and precision at the same time. This research also shows the possibility of designing Amharic IR using linguistic technique. Therefor further research especially on searching part of linguistic approach of Amharic IR would yield even better result.

CHAPTER ONE INTRODUCTION

1.1. Background

The long history of Information Retrieval (IR) does not begin with the internet. Prior to the broad public day-to-day use of search engines, IR systems were found in commercial and intelligence applications as long ago as the 1960s. The earliest computer-based searching systems were built in the late 1940s and were inspired by pioneering innovation in the first half of the 20th century [**1**]. The emergence of the term information retrieval (IR) was back in 1951 by Calvin Mooers (1951). IR definition by Calvin Mooers is as follows [**2**]

"Information retrieval is the name for the process or method whereby a prospective user of information is able to convert his need for information into an actual list of citations (queries) to documents in storage containing information useful to him. It is the finding or discovery process (searching) with respect to stored information (corpus). It is another, more general, name for the production of a demand bibliography. Information retrieval embraces the intellectual aspects of the description of information and its specification for search, and also whatever systems, technique, or machines that are employed to carry out the operation. Information retrieval is crucial to documentation and organization of knowledge"

The other well-known definition of IR is given by Winogrand(1972) as follows, information retrieval (IR) system does not change the knowledge of the user on the subject of his/her inquiry. It merely informs on the existence (or non-existence) and whereabouts of documents relating to the user request.

One of the main reasons that waken up IR researches in 1960s is Information Overload. This term coined by Bertram Gross in 1964, which means the difficulty on understanding information for decision making, because there is too much information on a topic [**3**]. This topic gets more attention when Alvin

Toffler published his bestselling book called the FUTURE SHOCK in 1970. With this the need of information retrieval becomes very vital.

In the field of librarianship, the way that items were organized in a collection was a topic. The classic approach was to use a hierarchical subject classification scheme, such as the Dewey Decimal system, which assigned numerical codes to collection items. However, alternatives were proposed by Taube [**4**], which was essentially a proposal to index items by a list of keywords. A few years later, Cleverdon [**5**] conducted a detailed comparison of retrieval effectiveness using Uniterms and the more classic classification techniques. His conclusion that Uniterms were as good as and possibly better than other approaches caused much surprise and his work came in for extensive experiment; results were found to be correct, and as a result, the use of words to index the documents of an IR system became established. Many aspects of Cleverdon's test collection approach to evaluation are still used in both academic research and commercial search testing today.

With the advent of computers, a great deal of thought has been given to using them to provide rapid and intelligent retrieval systems. In libraries, many of which certainly have an information storage and retrieval problem, some of the more boring tasks, such as cataloguing and general administration, have successfully been taken over by computers. However, the problem of effective retrieval remains largely unsolved [**6**].

It is when vast amounts of information to which accurate and speedy access is becoming ever more difficult. One effect of this is that relevant information gets ignored since it is never uncovered, which in turn leads to much duplication of work and effort [**6**].

Today Information Retrieval (IR) is a technology that involves computer-based retrieval to locate information that is relevant to a user's query. Typically this system searches in collections of unstructured or semi-structured data [**1**]. IR is the process by which a collection of data is represented, stored, and searched

for the purpose of identifying relevant documents as a response to users' query [**7**].

IR process involves two major sub parts; indexing and searching. Indexing is a process of creating logical representation of documents that are found in the corpus. It is an offline process of organizing and representing large document collection using index term. On the other hand, Searching is the online process that maps users' information need (represented in the form of query) with the documents representation(index files) by using matching methods and returning relevant documents from the collection of unstructured or semi-structured corpus to satisfy users' need [**8**].

Processing of natural language for IR systems is a bit challenging task. As of now there are no NLP techniques that allow extracting a document's or query's meanings without any mistakes. There are two approaches that are used to apply NLP techniques in IR: a statistical approach and linguistic focus [**9**].

The first is the classical model of information retrieval systems; which is characterized from each document's set of key words, known as index terms. This is a very simple focus based on the "bag of words". In this approach, all words in a document are treated as its index terms. Moreover, each term is assigned a weight in function of its importance, usually determined by its appearance frequency within the document.

The second approach is based on the application of rules that explicitly encode linguistic knowledge [**10**]. The documents are analyzed by linguistic tools that incorporate each level's own annotations to the text. The morphological analysis is performed by taggers that assign each word to a grammatical category according to the morphological characteristics found. The aim of this technique is to perform superficial parts of word analysis to identify the semantics of the word in its inflections.

IR continues to evolve as the computing environment changes. The most obvious recent example of this type of change is the rapid growth of mobile devices and social media. One response from the IR community has been the development of social search, which deals with search involving communities of users and informal information exchange. New research in a variety of areas such as user tagging, conversation retrieval, filtering and recommendation, and collaborative search is starting to provide effective new tools for managing personal and social information [**1**]. Also there are efforts of applying NLP in IR

The recent year efforts of integrating natural language to the field of IR, which have improved the effectiveness of the retrieval systems in languages like Chinese, Arabic and English also. This allows the processing of huge amounts of textual information with an acceptable level of efficiency in English. An example of this is the application of these techniques as an essential component in web search engines, in automated translation tools or in summary generators [**11**].

Despite the state of the art of IR in English and other Asian languages like Chinese and Korean, Amharic IR is still at its infancy stage. Some of Amharic IR design attempts are Stemmer by Nega, automatic classification by Zelalem, Application of IR on the web by Saba and N-gram indexing approach by Bethlehem [**12**]. Notably all these study use already available models that are built for some other language.

1.2. Statement of the Problem

Amharic is catching fire in producing digital documents. It reach the level where having effective working IR system is mandatory. Most of the previous attempts of designing Amharic IR are focusing on the model they are using than the language [**13**,**14**,**15**,**12**]. Due to that the problems related to the language behavior for constructing Amharic IR are remain unsolved.

Amharic's characterstics are different from that of English in many ways. Because of this there is a challenge in applying models of English language at

different stage of preprocessing. The major challenges of Amharic IR due to its linguistic characterstics are Inflectional word formation, Linguistic variation and linguistic ambiguity.

By inflectional word formation it is to mean that most of the words are the result of inflection from a root term. That means the suffixes and prefixes that cause inflection are not mere apex, a single sentence in English can be represented in one inflected word in Amharic. Table 1-1 shows the number of inflection from the root term due to suffix.

ልጇ ልጇም ልጇና ልጇን ልጇንም ልጇንስ ልጅ ልጅህ ልጅህም ልጅህና ልጅህን ልጅህንም ልጅም ልጅስ ልጅሽ ልጅሽስ ልጅሽን ልጅቼንም ልጅነት ልጅነትና ልጅና ልጅን ልጅንም ልጅዋ ልጅዋም ልጅዋን ልጅዋንም ልጆቹ ልጆቹም ልጆቹና ልጆቹን ልጆቹንም ልጆቹንና ልጆቹዋም ልጆቻቸን ልጆቻቸው ልጆቻቸውም ልጆቻቸውና ልጆቻቸውን ልጆቻቸውንም ልጆቻቸውንና ልጆቻችሁ ልጆቻችሁም ልጆቻችሁና ልጆቻችሁን ልጆቻችሁንም ልጆቻችን ልጆቻችንም ልጆቻችንን ልጆቻችንንም ልጆቻችውን ልጆቼ ልጆቼም ልጆቼና ልጆቼን ልጆቼንም ልጆች ልጆችህ ልጆችህም ልጆችህና ልጆችህን ልጆችህንም ልጆችም ልጆችሽ ልጆችሽም ልጆችሽን ልጆችሽንም ልጆችና ልጆችን ልጆችንም ልጆችንና ልጆችዋ ልጆችዋም ልጆችዋና ልጆችዋን ልጆችዋንም ልጆዎ ልጆቿን

Table 1-1: Linguistic variation of the word ልጅ with postfix inflection

From this simple word derivation we see that averagely used Amharic words can decline up to 80 different related words from addition of suffix. For real life applying statistical preprocessing will definitely fail because predicting and putting in table the possible affix lists is not practical [**14**,**15**,**16**].

An alternative solution for inflectional word formation might be free text indexing [**8**] yet it did not for Amharic, because the construction of free text indexing need exponential time and space.

The other challenge related with Amharic IR is linguistic variation we mean the possibility of using different words or expressions to communicate the same/related idea. Linguistic ambiguity is when a word or phrase allows for more than one interpretation [**9**]. Both situations affect the information retrieval process, even though in different ways.

Linguistic variation provokes document silence, that is, the omission of relevant documents that fulfill information needs, because the same terms were not used as those found in the document.

Linguistic ambiguity, on the other hand, implies document noise, or the inclusion of non-meaningful documents, since documents were retrieved that used the same term but with a different meaning.

Consider Amharic sentence "አበበ ለገና አዲስ አበባ አልመጣም።" with lemmas ገና, አደስ, አበበ, መጣ

- አበባ and አበበ are linguistic variations of the same root term, even the term አልመጣም
- ለገና the is a linguistic ambiguity(polysemy) to the word ገና

With the conventional text preprocessing for indexing the resulting terms would be አበበ, ገና, አደስ, አበበ, መጣ. Since አበባ is considered as linguistic variation it is represented with the stem አበበ. This means all the documents that contain the term አበባ are silenced or given small weight. This may be a lot worst when linguistic variation is negation as the term አልመጣም. The root term would be the total opposite meaning with root word መጣ.

On the other hand the word ገና "Christmas" other meaning means "not yet". Retrieval of using the root, there is equal chance of getting words that mean "Christmas" as well as "not yet" this create a noise in the retrieved documents list.

A good information retrieval technique should include the linguistic characteristics. IR models that are working well on English might not able to apply it on Amharic effectively. Because all models of IR approaches is not suited for all linguistic behaviors of languages. From the beginning statistical models were not designed considering Amharic language characteristics in mind. [**18**] [**19**]. Since Amharic differs from English in phonic level, word level, even grammar level than other languages it needs its own way of IR approaches that consider the linguistic approach [**17**].

Accordingly, Amharic IR systems there should incorporate basic characteristics of the linguistic while designing the retrieval system. These techniques should

try to come up with preprocessing, indexing and/or weighting technique that answers the problem of morphological variations, linguistic ambiguity.

To achieve this, the following research questions are answered at the end of the thesis.

- What are the core linguistic features of Amharic that should be taken into account for designing linguistically motivated Amharic IR system?
- Ho to design preprocessing technique that addresses the challenge of inflectional word formation?
- How to design indexing structure that controls linguistic ambiguity and linguistic variation of Amharic language?
- To what extent linguistic motivated IR technique improve the traditional word preprocessing?
- To what extent linguistically motivated IR improve the effectivess of statistical Amharic IR?

1.3. Objectives of the study

1.3.1. General Objectives

The general objective of this research is to design IR system by using linguistic approach of preprocessing, indexing and searching on Amharic documents.

1.3.2. Specific Objectives

To accomplish the general objective of this research, the following specific objectives should be met.

- To review literatures on IR, NLPs contribution in IR and Amharic IR system design attempts.
- To study in detail the linguistic features of Amharic words and its writing system.
- To prepare a document corpus that is suitable for designing Linguistic based retrieval system.
- To design Linguistic preprocessor based on the linguistic rule and features of Amharic word formation.

- To develop a new indexing approach that resolves the issues of linguistic ambiguity and linguistic variations of Amharic language.
- To develop a customized approach of term weighting that that determine the importance of a term in the Amharic documents.
- To evaluate effectiveness of the linguistic word preprocessor and compare it with the statistical techniques.
- To evaluate the performance of the retrieval system using performance evaluation techniques.

1.4. Scope and limitation of the Study

Scope of this research is to design a Linguistically Motivated IR (LM-IR) model for Amharic language. This model will be shown by developing a prototype experimenting purpose. The results from experimentation will be further compared with previous Amharic IR systems that were designed using statistical approach.

This research does not apply statistical IR preprocessing techniques like, stemming, normalization, and stop word removal. Rather, uses self-developed technique based on Amharic language called Linguistic Analyzer (LA). The LA differs from the statistical approach with preprocessing, indexing, weighting and matching phases. LA is python program with the concept of regular expression that could perform Morph Synthetic Analysis (MSA) of words with the corpus based approach. The index file structure used is called multilevel LMI; this index file is build up on the inverted file structure type.

The corpus for experimentation is limited to documents that are collected from Amharic holy bible. The experimentation LA of LM-IR has about half a million words from 1180 collection of documents. Unavailability of credible collection of text and the time require in preparing more than what is done is very time consuming.

1.5. Methodology of the Study

To achieve the objectives of the study and answer the research questions this research follows experimental approach to show the validity of the linguistic

approach. The following steps are followed with the research; corpus preparation, design and test different Algorithms on the corpus. The steps followed are

1.5.1. Literature Review

Previously proposed related literatures from books, articles and conference proceedings are briefly reviewed in order to have detail understanding on the advancement of IR, contribution of NLP in IR and how the Amharic IR research attempt evolved. A through study also made on the behavior of Amharic word formation and inflection types. The Ethiopic/ Amharic word representation relating on other Unicode encoded languages studied in detail in order to program the proposed model effectively using python 2.7.X programming language. Different techniques and Algorithms are studied, to get a suitable model for Amharic retrieval system that is linguistically capable of analyzing the word inflections.

1.5.2. Corpus preparation

The main content of the corpus for designing and experimenting the proposed LA and retrieval system is collected from all the 66 books of Amharic Holy bible. The Bible version is the year 1956 printed by Birhana Selam Printing Press. But the collected data is from electronic format of bible software called Iota.

The organization of the corpus is in order of their natural position in the bible. The chapters in the whole bible are organized in the respective chapter name and the each book in the bible has a folder for the chapters inside. The LA performed the analysis on this corpus for building Linguistic corpus (LC) and Linguistically Motivated Index (LMI) file.

The corpus preparation was assisted with the Python program to enable effective file structure for further processing. This enable to store the file names of files in the corpus in the list format that would enhance the retrieval performance during searching.

1.5.3. Implementation Tools

Now a days there are plenty of object oriented programming language choices for delivering some project, but, these languages show a difference in performance for a difference projects. After considering the requirement from this research that should be delivered, Python is given a preference of choice programming language.

Python is a high-level open source language that continuously extends its influence, from small development projects to the enterprise. It powers the highly respected Zope, TurboGears, and Django frameworks. Python advantage for Amharic retrieval systems is based on the following reasons.

Advantages of Python over other are programming Languages?

- ✓ *Text Operation:* It is high level object oriented programming language with excellent capability of text and file manipulation that make it suitable to handle text retrieval system. This research uses Python programming language for codding the proposed system. Python is used because of its capability to handle Unicode characters the same way as ASCII with small steps. Unicode is the encoding of Amharic writing system.
- ✓ *Cross platform:* python executed virtually on every machine available today. Windows with 32 and 64bits. MACs and Linux. Even it runs on some mobile computing Medias like tablets and smartphones.
- ✓ *Support from the programmers' community:* this programming language it has got greater support from the programming community all over the world.
- ✓ *Simple to learn:* python is simple for leaning to code. Natural language style of code writing makes it easier to learn the codding the meaning of the program.
- ✓ *Integration:* python programming language could easily integrate with deployment tools for web, database and desktop application

implementations. Python IDEs play a major role in integrating with other programming kits for deployment.

The version used is Python 2.7.X, this python version is a well-known python by many programmers for its codding simplicity. Many applications on the web and desktop are built by using this version. In addition this version has the following advantages

- The programmers community familiar with this version
- Enough support from online community
- Application written using this version usually runs without bugs

Python Integrated Development Environments (IDE)

Python IDEs provides tools for rapid and iterative development to ensure good Python development. [**20**]. There are more than ten IDEs; this number of different IDEs makes it clear that there is no single best Python IDE. The range of problems they attempt to provide solutions to be too wide to allow a single best IDE choice. The IDLE is the default developer environment that we get while we download python from www.python.com. It is very simple and clear text editor, which uses color codding for keywords and methods that are found in the python classes. Its error tolerance makes it preferable while designing simple modules of the big programs on the fly.

1.5.4. Testing Procedure

There are two systems that need to be tested in this research; the first is the performance of LA, the second is the performance of Linguistically Motivated IR system (LM-IR).

LA is the linguistic analyzer of words for preprocessing before indexing document words or before weighting query terms. LA performance means, how effectively the LA preprocessor groups the words in the corpus to their correct root terms in spite of different inflections of the term. This performance is

further compared with the statistical preprocessing approach of stemming that uses prefix and suffix removal technique.

The performance of preprocessor for both LA and statistical techniques is measured by using accuracy. Accuracy calculated the percentage of correctly classified words from inflections of the same root.

To calculate LM-IR performance, first relevance judgment matrix constructed for the queries against the document corpus build for this research. Recall and precision are used to evaluate the performance of the designed IR system. Recall is how much of the relevant documents are retrieved. Precision is how much of the retrieved documents are retrieved. F-measure is the harmonic mean of Recall and Precision. In addition F-Measure will help us to find the harmonic mean of both precision and Recall.

1.6. Significance of the Study

This research creates a new dimension of linguistic integration for the development of Amharic IR. Below are the major other significances of this research.

1. New approach of IR design in the local language:
    - The previous retrieval researches [**13**,**14**,**15**,**12**,**21**] on Amharic IR consider model before the language. For that reason most of the effort is how to get better result using already existed models for other language than to design one that is suitable for the linguistic of Amharic. Due to that there is no significant attempt to model the linguistic of Amharic into IR for better retrieval model design.
    - But on this research started from the linguistic features of Amharic language and try to find (design) a model that is suitable to handle it.
2. Explore effective NLP integration into Amharic IR:

- In early 90s NLP has been criticized for its adverse effect on the performance of IR system especially on efficiency [**22**]. But this study effectively integrated LA, LM Index file structure and LM term weighting technique in the design.
- This research will create a new dimension in developing Amharic information retrieval system by incorporating Natural Language Processing (NLP) technique on the indexing, weighting and matching models. All the previous researches adopt popular IR indexing and matching models. These models are purely nonlinguistic approaches towards designing the Amharic IR systems. The indexing, matching, and weighting techniques applied are new for the local information retrieval designing effort. This research tried to find the importance of NLP in designing Amharic retrieval system. It gives a sense of semantic way of Amharic IR system design.

3. MSA for preprocessing:
    - Morph syntactic Analysis (MSA) for preprocessing of terms in the corpus proved to outperform the statistical stemming technique. This is due to the MSA is designed based on the core linguistic behavior of Amharic language; those are syllabic writing nature and morphological word formation.
4. Term Inflection degree(Id) for term weighting:
    - This behavior is based on the linguistic feature of Amharic words that, if a word has a capability of inflecting in many different forms in the document, then there is more information related to that word [**17**].

There are also other significances of this research to Amharic IR. For instance the multilevel indexing is one of the best features to keep the semantic meaning of Amharic that caused due to preprocessing.

1.7. Organization of the study

This study is organized in to six chapters. The first chapter discusses introduction that provides background information, the aim of the research, its objectives, research question, methodology, scope and limitations and significance of the study.

Chapter two presents reviews the literatures on IR, the need of IR, the contribution of NLP in IR and various techniques, models and methods of information retrieval system.

The third chapter discusses about Amharic language history, the development of its writing system, computer encoding approaches to the languages alphabets. This chapter also discusses the main types of Amharic word formation and its challenges.

Chapter four of the study is about the design of linguistically motivated IR. It discusses about the limitations of the statistical IR word preprocessing in detail and compares it with the linguistic approach of IR design. This chapter shows the designed models for LM-IR, the approach followed the Algorithms needed to implement it and even the python code related to the Algorithm that make it happen in the experimentation model.

Chapter five shows the experimentation of chapter four's design it discusses the corpus preparation and experimentation on LA and LM-IR system. The final chapter, chapter Six presents the conclusion drawn from the findings of the study for both LA and LM-IR and recommendations that should be considered in future researches for designing better linguistic Amharic IR.

CHAPTER TWO LITERATURE REVIEW

The general theory of Information retrieval is to place the user and his information need closer to the source in the searching process. The theory presupposes a similarity in use of terminology, concept relations, between generators (authors) and potential searchers [**8**]. The objective of Information retrieval is to bridge the gap between the authors' world (those who generate/create information content) and the readers' world (those who looks for information) by building effective communication (IR).

IR concerned with the process involved in the representation, storage, searching and finding information which is relevant to a requirement for information desired by a human user [**23**]. In IR environment a successful retrieval approach is that is able to provide the most relevant results to the user in a conceivable amount of time [**24**]. The outcome of IR process is a set of documents containing information on a given topic. These documents may be in different format, text, audio, video or a combination of those [**25**].

Any IR systems start with data (document/knowledge) representation and ending with returning results to the user. The intermediate processes involve searching and match operations, ranking mechanisms, and filtering process [**24**].

Figure 2-1: Basic IR System process Model

Figure 2-1 shows the basic IR process of model. This process first represents both users' information need and available Information. The matching model then uses to map between these two information needs.

2.1. Information Representation

There are two major ways of information representations, the natural language approach of Information representation and Controlled Vocabulary (non-natural language) information representation approaches. Fig2 shows the information representation by the degree of their departure from Natural Language [22,26].

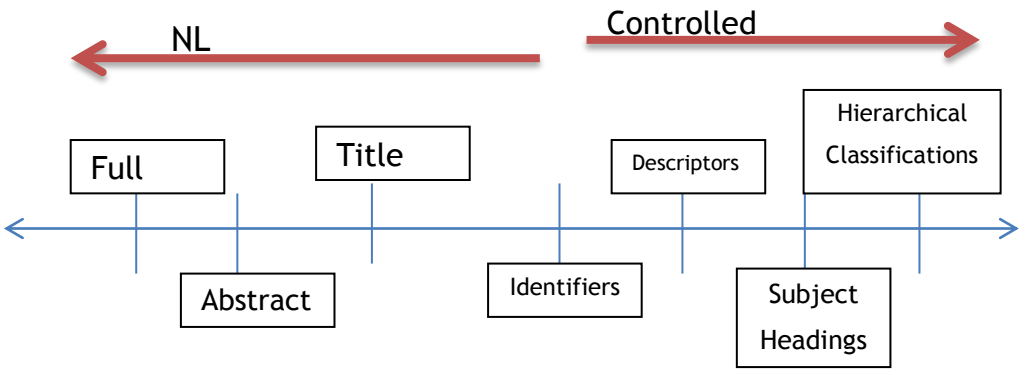


Figure 2-2: Information representation languages, arranged by their departure from natural language

In the Figure above the left half presents the natural language approaches to information representation and the right half part presents the controlled vocabulary approaches to information representation. The natural language approaches include full texts of documents, abstracts, titles, and identifiers extracted from the original text by indexers. The controlled vocabulary approaches include descriptors, subject headings, and hierarchical classification.

The difference between identifiers and descriptors is that identifiers are derived from the original text, whereas descriptors are listed in thesauruses, which helps to deal with synonyms, homographs and such. The difference between descriptors and subject headings, on the other hand, is that thesauruses are usually derived from existing document collections, whereas subject heading lists are often attempts to represent the whole structure of universe instead of representing the vocabulary of specific document collection [26].

Information representations goes from identifiers to hierarchical classification represent more of meta-information, but, full texts, abstracts, and titles represent the potential information itself. That means the classes show the degree of meta-information and potential information. At the left edge, ('full text') the degree of potential information representation is highest and toward the right edge the degree of potential information decreases. On the other hand, the degree of meta-information is highest at the right edge ('hierarchical classification') and decreases toward the left edge.

It is obvious that a full text contains the highest degree of potential information, because it contains it all. Naturally, full texts include all the identifiers that represent meta-information as well. However the degree of meta-information is the lowest. Hierarchical classification, on the other hand, may be considered to represent the highest degree of meta-information and the lowest degree of potential information, since its descriptions are most general and standardized [**27**,**8**].

2.2. Indexing

American National Standards Institutes (1968) defines; Index is a systematic guide to items contained in, or concepts derived from, a collection. These items or derived concepts are represented by entries in a known or stated searchable order, such as alphabetical, chronological, or numerical [**28**]. This process called indexing.

An index term is a keyword expression which contains a considerable amount of information (or meta-information) about the content of a text; this term is used to depict documents in a collection. This means documents in a collection are represented by the index term [**29**].

Indexing is the process of analyzing the informational content of records of knowledge and expressing the informational content in the language of indexing system [**26**]. The indexing system is the set of prescribed procedures

(manual and/or machine) for organizing the contents of records of knowledge for purposes of retrieval and dissemination. The goal of Indexing is to enable fast retrieval during searching while doing processing of words in the document offline by using indexing structure. Indexing is the complement of searching in IR that used to speed up access to the desired information in the document that is found in the corpus as per users' query. The other advantage of indexing is save storage space and improves the performance of retrieval system, since only vocabulary files are uploaded to the primary memory [**18**,**30**,**26**].

Every indexing process involves the follows two steps, the first is to select indexable word (concepts) from the documents in the corpus and the other is express these concepts in the language of the indexing system, that is index file structure

Indexing theory has in general developed around two concepts; Linguistic and statistical Indexing theories. Linguistic indexing also known as Natural Language Inherent Indexing (NLII) refers to the use of linguistic analysis for identifying concept (semantics) of words from their internal structure [**31**,**32**]. The statistical indexing is a controlled vocabulary indexing where words in the document are preprocessed with statistical techniques like tokenization, normalization, stemming before indexing [**8**].

Statistical indexing approaches are well known and used in the commercial and many of research approaches. There are many fully developed models for IR based on statistical indexing technique. On the other hand, linguistic approach of indexing doesn't get enough attention in the design and implementation of IR systems [**33**].

There are different types of different index file structures for statistical approach of indexing. The notion of index-term-structure is a kind of content analysis framework for information retrieval. Linguistic approach tried to

provides evidence for determining the index-term-structure. There are no well-developed linguistic approaches of index file structure for NLII [**6**].

2.2.1. Statistical Indexing Structure

None linguistically (controlled vocabulary/ Statistical) indexing suggest a string of predefined terms or keyword phrases that typically represent the indexer about-ness of a document. This indexing structure is currently powering commercial online retrieval [**34**]. The common and well known controlled vocabulary indexing structures are Inverted file, sequential files and signature file [**35**].

Sequential File,

Sequential file is straight forward way of indexing of files/documents sequentially. This indexing structure put records one after another [**36**], it does not need vocabulary as well as linking pointers. After documents are preprocessed and content bearing terms identified they are arranged serially, one after another [**30**].

Sequential files can build can be built relatively simpler compared to other index file structure. Since content bearing words in sequential file are arranged in sorting order the access time required is *O(nlogn),* where n is the total number of content bearing terms in the corpus.

The main challenge of sequential indexing structure is index file updating. Figure 2-3 shows the main limitation of Sequential File indexing structure is its lack of incremental indexing. If a new word needs to be indexed, it has to be in sorted order, which force displacement of the total words below the new word if the file arranged in ascending order. This process is equivalent to re indexing of the whole index files again and again every time a new word appears.

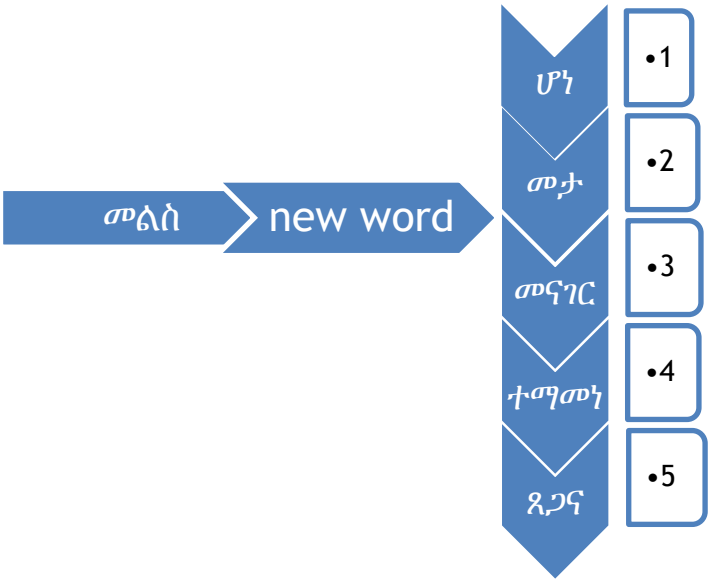


Figure 2-3: sequencial index file update challenge

Inverted file

Inverted file structure is the well-known structure that is currently used with the major search engines, general purpose mainframe based database management systems commercial and research based retrieval systems to speed up searching task [**37**]. Inverted files comprises of two parts, vocabulary file and posting files.

Vocabulary file also known as lexicon. The lexicon piles unique occurrences of words in the corpus/ language. Each entry in the vocabulary has the word and a pointer to the posting file. The other information a vocabulary file entry has is the number of documents that have the lexicon and collection frequency.

Heaps law shows the relationship between total numbers of words with relation to the vocabulary terms. To answered how much the vocabulary would be, According to Heaps' Law $V=O(nB)$, where, $0.4<B<0.6$. With this calculation 1GB of text documents have only 5MB of vocabulary file size. The construction of vocabulary file will be at most one per occurrences of the word in the text $O(n)$.

Posting file is the file structure that maps the vocabulary entry with its significance in the corpus. It has a document id to identify the document, count of vocabulary term in the document, a link actual file and a reference to the locations of the term. Accordingly the inverted search Algorithm starts looking the query words in lexicon, then follow the pointer to the posting file and manipulate posting file according to the retrieval model applied [**38**].

Signature file

A signature is a bit mapped abstraction of record. There are two types of signatures; word signature and sequential signature files. Signature file works based on hash coded. It is a word oriented index structure [**39**].

Word signature approach identifies words or n-grams of the record that are hashed to bit patterns, word signatures are letters which are concatenated to form the record signatures supper imposed codding methods hashed each unique identifications to S bit position in each bit string with fixed width F and supper imposed the resulting signature. In word signature with there is a chance that two different n-grams will hash to the same bit position. This is referred to a collision. Since we choose F much less than the total number of unique n-grams, collision are possible.

A sequential signature file, or SSF is made up of serious of signatures one for each record to be indexed. A query is processed by generating its signature file via the same process used to generate index signatures and comparing it to each signatures in SSF by bitwise AND. Then the matching records are retrieved using an address table. Even after retrieval the documents should be compared to the query because some of them can be false matches due to collision [**40**].

Suffix Tree or Suffix Arrays,

Suffix tree or Suffix array is a sorted list of suffixes of given string in lexicographic order [**30**]. If a suffix of the string is a prefix of another longer suffix, the shorter suffix must end in an internal node instead of a leaf, as desired. It is to avoid this possibility that the unique termination character is

added to the end of the string. We first show that the suffix array and Longest Common Phrase (LCP) array of a string can be obtained from its suffix tree in linear time.

- ✓ Lexicographic ordering of the children of a node to be the order based on the first character of the edge labels connecting the node to its children.
- ✓ Lexicographic depth first search to be a depth first search of the tree where the children of each node are visited in lexicographic order.

The order in which the leaves of a suffix tree are visited in a lexicographic depth first search gives the suffix array of the corresponding string. In order to obtain LCP information, the string-depth of the current node during the search is remembered. This can be easily updated in O(1) time per edge as the search progresses. The length of the LCP between two consecutive suffixes is given by the smallest string-depth of a node visited between the two suffixes. The improvement of suffix arrays over suffix trees includes the space requirements, simpler linear time construction algorithms and improved cache locality [**41**,**42**].

2.1.2. Linguistically Motivated Indexing (LMI)

LMI is a NLII technique that focuses on analyzing of documents to identify concepts that could represent the document in the resulting index structure. This indexing is meaning representation by constructing the words semantic variation. The Algorithm produces a range of alternative syntactic expressions for all the forms of each concept. Indexing concepts used to search document file identification [**31**,**32**].

LMI resolves the challenges related with statistical indexing [**26**]. The limitation with the statistical approach indexing is its incapability to handle linguistic variation and Morphological ambiguities. This hinders the designing of

generic indexing and weighting technique that manage the linguistic related problems [**9**].

The goal of LMI is designing to achieve a good and flexible indexing by identifying index term source in the meaning representations built by a powerful general purpose analyzer [**31**].

Amharic IR that is managed with LMI systems is getting the advantage of semantically handling the linguistic variation and linguistic ambiguities. Since NMI indexing technique stemming heavily affect the retrieval performance because Amharic is heavily morphological language. Semantic meanings of words are lost during the stemming process. Applying LMI improves the retrieval performance.

Till today there is non-linguistic way of indexing the Amharic Words. This research used parts of word of Amharic words to build semantically motivated index terms and customized Vector space model and weighting technique. The detail of the linguistically motivated indexing of the Amharic words is mentioned next chapter.

2.3. Matching Models

Searching is the process of relating index terms to query terms and return relevant hits to users query. There are different models used for designing IR systems. The choice of the model is depend on the purpose of the retrieval system. Below are well known matching models.

2.3.1. The Boolean Model

A simplistic way of matching users' information need with index structure; which have represented in Boolean expression. This matching function considers the document as weather relevant or irrelevant with relation to the absence and presence of particular term. Retrieval is based on binary decision criteria. This model states weather a particular document is relevant to a term or not relevant, no weighting applied. The document is associated with a set of

keywords. Terms in the users information need can be concatenated with logical operators AND, OR, or NOT/BUT to further refine user need. [**18**,**30**].

The advantage of the Boolean model is that it gives control for the user over the system what to retrieve or not. This process makes query reformulation simple since users could decide what to display or not.

The disadvantage of Boolean model is that it could not give ranking for retrieved documents. For the case where a term is matched to all the documents in the corpus it retrieved the whole docs. On the other hand if the term did not match with any of the documents in the corpus it returned null [**18**].

2.3.2. The Probabilistic Model

The goal of probabilistic model is to retrieve documents in order of their probability of relevance to the query. The model works by calculating the probability of getting relevant documents from the retrieved list of documents. It is based on the query of user the documents are categorized in to relevant documents or non-relevant documents. The user then observe the first retrieved documents and gives feedback for the system by selecting relevant documents as It is the estimation of the probability of relevance that a document *Di* will be judged relevant by the user with respect to query *q*. which is expressed as, $P(R|q,Di)$ [**14**,**8**,**43**];

Where, *R* is the set of relevant document.

Probabilistic model is dependent of user's feedback in the retrieval process of reviewing what the system provides as a relevant. A reference retrieval system's response to each request is a ranking of the documents in the collection in order of decreasing probability of relevance. However, the probability of any document is decided by the system since it is unknown for the first time. Therefore, the probabilistic model needs to guess at the beginning of searching process. The overall effectiveness of the system to its

user will be the best that is obtainable. This model is based on user's relevance feedback to improve the retrieval performance, the assumption that there is a set of documents that satisfies the users information need expressed in the query [**14**,**43**].

1.2.3. Vector Space Model (VSM):

This model vectors terms in the documents and query in n-dimensional vector. VSM arranges all possible content terms in the document and in the query by assigning term weights [**44**]. The weighing technique usually follows tf*idf. It is a composite term weighting factor that obtain both recall and precision in the enhancing.

Term frequency is the simplest approach to assign the weight to be equal to the number of occurrences of term *t* in document *d* represented as $tf_{t,d}$. term that are frequently mentioned in individual documents appear to be useful in recall enhancing. This factor used as part of the term weighting system measuring frequency of occurrence of terms in the document or query texts.

Term frequency represented as

$$W_{t,d} = \begin{cases} 1 + \log tf_{t,d}, & if \ \ tf_{t,d} > 0 \\ 0 & otherwise \end{cases}$$

Idf-Raw term frequency as above suffers from a critical problem: all terms are considered equally important when it comes to assessing relevancy on a query. In fact certain terms have little or no discriminating power in determining relevance even if they appear frequently in the document. To this term frequency factor alone cannot insure acceptable retrieval performance. Specifically, when the high frequency terms are not concentrated in few particular documents, but instead prevalent in the whole collection, all documents tend to be retrieved and this affects the search precision. To identify only terms concentrated in in only few documents of the collection inverse document frequency (idf) factor performs this function. The idf factor

varies inverse with the number of documents n to which a term assigns in the collection of N documents [**44**,**45**].

A typical idf factor may computed as $idf = Log\frac{N}{n}$

The same way the Tt*idf weight calculated as

$W = (1 + \log tf_{t,d}) * (\log {}^{N}/_{n})$ Where,

- ✓ $tf_{t,d},\ is\ the\ number\ of\ term\ t\ in\ document\ d$
- ✓ $N, the\ total\ number\ of\ documents\ in\ the\ corpus$
- ✓ $n, the\ number\ of\ documents\ that\ hold\ term\ t$

The retrieval function compares the query vector to every single row that represents a specific document in the vector space. In general sense to increase the weight the term frequency of term *i* in the document *Dj* have to be higher and term *i* should be less redundant in the overall collection of the corpus.

The well-known proximity measurements in VSM are distance and angle measurements. The proximity is similarity of vectors or inverse of distance. The purpose of proximity is to rank more relevant documents at the top [**14**,**8**].

Euclidian distance is the well knows distance similarity measurement. This measurement prioritize inverse of the distance between the query and the document. This similarity measurement did not perform well when for different length of query and documents.

In the Figure 2-4, the distance measurement will favor D1 and D3 to query Q than D2, even though D2 and Q have more similar distribution of terms. Since D2 contains more terms than the query term Q. it clearly show that distance proximity measurement fails to address when it comes the real time computations of similarity.

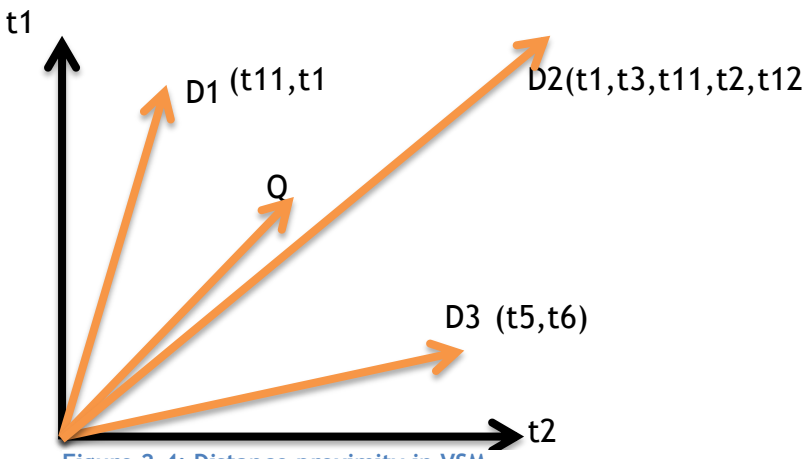


Figure 2-4: Distance proximity in VSM

Angle measurement is preferred way of measuring the similarity during the retrieval for ranking method. It measures the angle between the documents and the query. The angle measurement is done by cosine of angle between the document and query vectors. For length-normalized vectors, cosine similarity is implying the dot product (or scalar product):

To avoid the bias caused by different document lengths, a common way to compute the similarity of two documents is using the cosine similarity measure. The inner product of the two vectors (sum of the pairwise multiplied elements) is divided by the product of their vector lengths. This has the effect that the vectors are normalized to unit length and only the angle, more precisely the cosine of the angle, between the vectors accounts for their similarity.

$$sim(d1, d2) = \frac{\vec{v}(d1).\vec{v}(d2)}{|\vec{v}(d1)||\vec{v}(d2)|}$$

$sim(d, q) = \frac{\vec{v}(d).\vec{v}(q)}{|\vec{v}(d)||\vec{v}(q)|}$, since queries are considered as a short document.

Documents not sharing a single word get assigned a similarity value of zero because of the orthogonally of their vectors while documents sharing a similar vocabulary get higher values (up to one in the case of identical documents). Because a query can be considered a short document, it is of course possible to create a vector for the query, which can then be used to calculate the cosine similarities between the query vector and those of the matching documents.

Finally, the similarity values between query and the retrieved documents are used to rank the results.

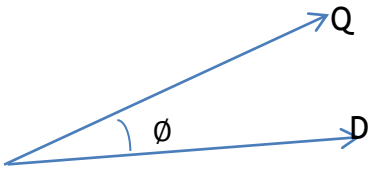


Figure 2-2: Cosine similarity formula

The similarity calculated by using the cosine $Sim(D,Q) = Cos(\emptyset)$

If $Cos(\emptyset)$ is closer to 1, which means both the query and the document have more similar terms. But, if $Cos(\emptyset)$=0, that means the query and the documents do not have any similar term.

Can be put this way using the weights of each term

$$Sim(D,Q)\frac{\sum w_{i,j} \times w_{q,j}}{\sqrt{\sum_j w_{i,j}^2}\sqrt{\sum_j w_{q,j}^2}}$$

2.4. The contribution of NLP to IR

In the year 90s, Integrating NLP with in IR has been criticized in English language. Because the research attempts of those times to integrate NLP with IR deliver non improved, sometimes disappointing result. Primarily the intention of integrating NLP technique for IR comes to resolve the problem of syntactic and lexical ambiguities of the language [**46**,**47**].

There are no enough attempts done to design a linguistic approach of IR systems thinking they will need of sophisticated Algorithm and high computing power to run them it will have negative results on the performance of the system because it requires higher-level processing during the linguistically analysis than the traditional IR preprocessing activities. [**48**,**25**]

Due to that most researchers and commercial IR systems prefer statistical methods over the linguistic approach while designing IR. Their argument is that NLP method increase processing and storage cost dramatically and could not be

applied to large collections [**48**,**25**,**32**]. The contribution of Natural Language Processing (NLP) in IR is has been ignored, it did not significantly study. Many researchers are not encouraging the integrating NLP techniques to IR.

One of the early disappointing results was reported by Smeanton [**47**] in the mid-1980s developed and experimented with techniques for parsing users' natural language queries and form the resulting parse trees. They identified word pair and word triples dependencies between query terms. Which were used as part of term weight retrieval, the performance improvement reported was negligible. There was also anther research performed by the same time by Fagan [**46**] and reported the same result.

The noted problem with the above model is that the syntactic level analysis generates too many noisy dependencies in both the query and the document. Realizing that researchers start to use a new model called Tree Structure Analysis (TSA). TSA is directly derived from morph syntactic analysis of input text based on constraint grammar framework [**49**]. If a text fragment has multiple syntactic interpretations in the TSA model, all of these are encoded into the TSAs and all are available at retrieval time with the TSA matching Algorithm weighting various interpretation of the text fragment.

Experimentation by modifying TSA matching is performed by Smeton [**50**] morph syntactic analysis documents by base form of words occurring within them. The researchers take top 1000 documents according to tf*IDF term weighting function and run TSA to TSA match between query TSA and document TSA. After ranking the documents for evaluation they report the effectiveness of TSA to be disappointing, much worse than tf*IDF ranking that used as pre-fetched. The reason they give for the poor performance is either the performance of syntactic analyzer or because they run the experiment on top of tf*IDF model ranked.

One thing that most researchers agree on is that NLP can contribute in resolving issues related with the linguistic ambiguities and linguistic variation

which are limitations of the retrieval system [**31**,**51**,**9**]. Therefore it's wise to strive on how to significantly optimize NLP Algorithms to increase performance than abandoning the role of NLP in IR. In addition to that NLP technique would become manageable by the time computing devices become more powerful. One example of this is WATSON. WATSON is a super computer about a size of medium house made by IBM. It uses NLP techniques during retrieval of information from its knowledge base. WATSON showed its capacity of natural language understanding by participating in Jeopardy (question answering computation) and beats two world champions. And now in this year (2013) there is ongoing project to make WATSON be available for consumers by miniaturizing it to laptops or desktop size.

The argument of expensiveness and computational power and memory storage utilization might not hold true in the future according to Moors Law, which states computing power will doubles every 12 to 18 months. The today's technology is much powerful than the time when NLP approaches in designing IR system were criticized for being expensive is expensiveness in the memory and computational cost to implement.

2.5. Related works

The effort so far to design Amharic IR is only experimental level. There is no commercial working IR system on Amharic language [**14**] [**15**] [**52**] [**53**]. All the previous attempts of designing Amharic IR use the already popular and widely applicable models of the English language, like N-gram by Bethlehem [**15**], latent semantic indexing by Tewodros [**52**] and application of WEBSOM for designing Amharic retrieval system by Bizuneh [**12**]. So far there is no notable attempt to design Amharic IR using linguistic approach.

Saba on application of IR technique on Amharic web documents [**16**], by Bethlehem Mengistu perform application of N-gram indexing for Amharic text retrieval [**15**], another indexing technique by Thewodros performed was by using Latent Semantic Indexing (LSI) approach with Singular Value

Decomposition (SVD) [**52**]. On the same year Bizuneh put WEBSOME as alternative for information retrieval techniques for Amharic text [**54**]. Recently Design and Implementation of Amharic Search Engine on Web Documents by Solomon shows a reported performance of, average precision 0.99, average recall 0.52 the F-measure 0.68 [**53**].semantic based query expansion technique by Bruk for Amharic IR yields 0.53 average precision, 0.73 average recall and 0.63 F-measure [**21**].

Bethlehem's [**15**] was an attempt to generate indexing structure for Amharic texts using N-gram based Automatic Indexing. She tried to solve the absence of standard index structure for Amharic language in IR. By applying, Bi-gram and tri-gram on 100 Amharic news articles, it is reported to be adversely affect the performance by increasing the processing and storage while generating index terms, and not efficient for large documents with higher n-grams. Hence, n-gram is not suitable for Amharic text indexing since it is highly morphological. On the other hand she suggested n-gram indexing for exhaustive indexing so that in improve recall.

Tewodros use latent semantic indexing approach for Amharic IR purpose. He compared latent semantic indexing with standard vector space model approaches. On his experimentation he is able to find 9556 unique words of Amharic from 206 news articles. He arranged the weighted matrix in standard vector space model and further computed with Singular Value Decomposition for latent sematic indexing. The result reported on from 110 dimensions using cosine similarity. When LSI methods compared with Standard Vector space model, the LSI showed shows better performance specially on the recall level between 0.9 and 1 [**52**]. Tewodros's conclusion was made on a very small number of corpus and experimentation model. His conclusion might not yield the same result on large amount of data.

Solomon and Tesema [**55**] try to design a full scale Amharic search engine. Their experimentation looks for documents written in Amharic and performs

preprocessing, indexing, weighting and retrieving. They obtain a precision of 0.99 and recall 0.55. The claim is that their system is doing different from pattern matching that Google and yahoo doing now. In the recommendation, incorporating more of the language's feature might improve the performance of the system.

Alemayehu [**13**] worked on query expansion for Amharic IR and Abey [**21**] also works in semantically way of query expansion. They tried to resolve the issue of polysemous and synonymous terms in Amharic by applying query expansion. On their experimentation they applied different techniques of query expansions; global analysis, local analysis, bi-gram analysis, bi-gram based thesaurus and statistical co-occurrence. Statistical co-occurrence method yields a better performance in the experiment 53% precision and 73% recall. On their recommendation is applying ontology based query expansion might further improve the performance of Amharic words that are polysemous and inflected.

Ammanuel [**14**] was able to experiment the effectiveness of probabilistic IR model for retrieving relevant documents from Amharic text corpus. He aims to design an applicable Amharic information retrieval system with improve performance by incorporating users relevance feedback. His probabilistic model showed 73% of F-measure by using Binary Independent Model (BIM) of the probabilistic approach [**14**]. He further suggested that incorporating thesaurus and co-occurrence analysis can further improve the performance of the Amharic IR systems.

So far there is no considerable attempt of integrate NLP with IR for Amharic, almost all studies were trying to use already popular models on other languages [**14**,**13**,**15**,**21**,**55**]. All of the local research approaches uses statistical (nonlinguistic) methods designing IR systems with linguistically approach.

CHAPTER THREE AMHARIC LANGUAGE

Amharic (Amarigna) language and its scripts are highly capable of information representation. It is the most commonly spoken language in the Ethiopia [**14**], by more than 32% of Ethiopian people uses it for their day to day activity of life. With population of 88,013,491, the world's 14th most populous country, there are significant numbers of the language speakers. Outside the country the language is spoken in other part of the world. Among these countries where the language is spoken substantially are USA, Israel, Egypt, Sweden and other European countries [**56**].

Amharic has been serving as a national language since 13th century in highland populations, who were dominant of the times. It was used in different areas of the government sectors, courts, religious organization and preparing official documents [**56**,**57**]. The current EPDRF government also uses Amharic as a national language in different government institutions and legal bodies and documents. The Orthodox and Catholic Churches also use Amharic language for sacred purpose and in scripting activities.

The main linguistic behavior of Amharic language that should be taken into consideration while designing any Amharic IR systems are; the syllabic nature of writing system and the morphological nature of word formation. This chapter covers the details of these two behaviors of the language in detail.

The language is scripting believed to be evolve from Ge'ez language. Ge'ez was serving is the class of semantic languages, which was serving as scripting language for Ethiopia literatures especially in Orthodox Church in the last centuries [**58**] [**14**].

3.1. The Development of Amharic Script Encoding

The Writing system of Amharic language called Amharic/Ethiopian script. Amharic language scripting is syllabic in which a character is used to represent a phoneme, which is a combination of a vowel and a consonant, presented as 2-

dimentional consonant-vowel combinations [**59**,**15**]. In Amharic scripts vowels and consonant are represent in as a single alphabet. Figure 3-1 shows sample traditional Amharic script from the Holy bible.

The Ethiopic/Amharic script is originally evolved for archaic language, Ge'ez. Ge'ez played a major role in the development and expansion of Amharic language and its writing system [**14**]. The scripting of Ge'ez language itself believed to be evolved from the sabian language, which is from south semantic group together with the Hebrew and Roman languages [**60**]. The ge'ez language has been serving mostly in the religious and poem in the church of Orthodox and royal families. For that reason the language could not serve for the community day to day activity. Therefore Ge'ez is now seized as the language of public communicable. Yet the Scripting of Ge'ez contributed for the development of Amharic/Ethiopic Scripting system [**60**,**57**].

Figure 3-1: Traditional Amharic Script

Amharic Script is currently used to write several languages in the country, including Amharic, Tigrigha and Oromiffa. It continues to be extended for writing language that have little tradition of printed typography; current characters to cover such extension may be added to the standard later as definitive information about them becomes available [**59**].

In the language, there are nearly forty characters which contains special feature representing labialization [**60**].

Figure 3-2:Amharic character inflections arrangement

The other set of characters are generated from this root characters by slight modification that caused by vowels added at the root alphabet. Figure 3-2 shows the inflection of Amharic alphabets from the root word. There are seven alphabet inflections from character root that are caused from the addition of vowel.

3.1.1. Encoding of Amharic Alphabets/Fidels

Amharic scripting is capable of representing any information in syllabic form. To enable usage of the script in computerized system, there has been an attempt of encoding Amharic scripts computer systems ever since the beginning of usage of computers at consumer's level before it reached the current representation of Unicode representation.

The first ever computerized scripting encoding was done on English language called American Standard Code for Information Interchange (ASCII). At the beginning ASCII was only encoding technique, which was created in 1963. This encoding works for English (Latin) alphabets and number characters only. In this encoding technique each character is a byte (8bits). With these bits we can only have 128 character representations [**61**].

ASCII encoding was unable to represent non English alphabets. Amharic letters were not also represented in the ASCII standard code. But Amharic is not the only language that lack scripting from ASCII. There were also other languages that were not encoded using ASCII text encoding standard that works for non-English languages that have their own scripting system like Chinese, Hebrew, Arabic Russian and other European language such as Germany, Spanish, and Greece.

Since then another alternative approach has been searched to encode non English alphabet and numerals. The encoding of Amharic characters passes through different stages. The pre Unicode encoding performed on Amharic was able only to recognize the minimal character sets those are 303 in number.

On screen encoding

These encoding techniques follow the postscript, True type or EGA/ VGA technique. The limitation of this traditional scripting technique is that it only represents a maximum of 256 characters in one font. That means it is not possible to store all the Amharic characters including ligatures, numbers and punctuation marks in one font. For that case the fonts has to be divided into two separate fonts as Ethiopic/ Amharic [**59**].

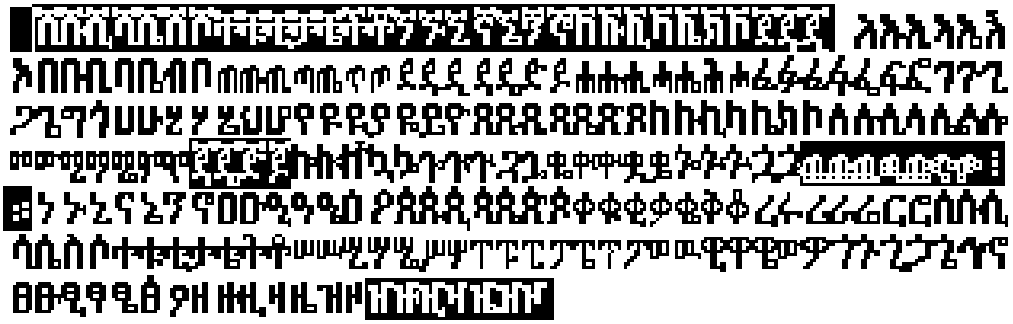


Figure 3-3: On screen encoding, VGA block (Amharic)

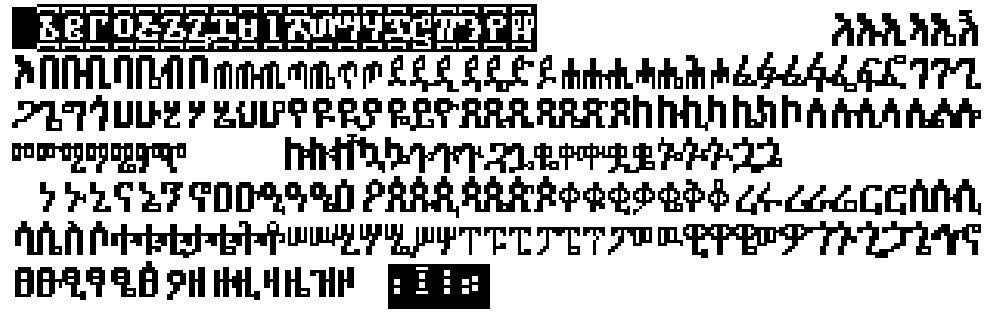


Figure 3-4: On screen encoding, VGA block 2 (Ethiopic)

VGA fronts were produced by using the screen pixels. The pixels were binary and can represent up to 256 (2^8) characters in a font. It was not able to put all the Amharic characters in a single font all Amharic characters, because there are more than 256 characters in Amharic language. Therefore they used two

blocks of font representations as it is shown in Figure 3-3 and Figure 3-4. By doing this it would be able to represent up to 256+256=512 characters [**62**].

The challenges of this kind of encoding is that if we are typing characters that are in font Amharic Figure 3-3 we could not access characters on Ethiopic Figure 3-4 without changing the font. This idea of interchanging fonts called "font switching" [**59**]

Character set encoding (Two byte encoding)

Character set encoding technique was by IBMS WordPerfect approach for font management. Besides one byte representation of ASCII characters there is a 2byte additional representation. This representation contains 12 characters set comprising a total of 1873 characters [**59**]

This representation puts Ethiopic and Amharic syllabic character sets in two places character set 11 and character set 12. Character set 11 holds Ethiopic and Amharic syllabic characters containing vowels ¯a, ¯ı, ¯u, ¯e, and ¯o Appendix VI have the complete set of character set 11 of Ethiopic and Amharic characters. The Amharic and Ethiopic syllabic characters in character set 12 are those who containing vowels 'a' and 'e'; numbers; punctuation mark. Appendix VII the complete set of character set 12 of Ethiopic and Amharic alphabets.

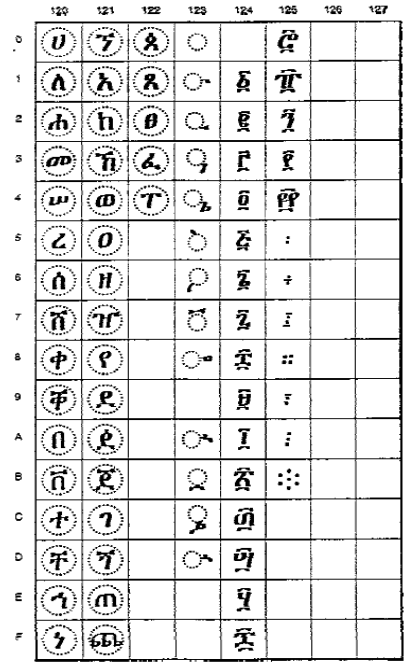


Figure 3-5: First proposal of Amharic to be Unicode encoded. 1200 to 127F

Unicode encoding

Unicode encoding is a full two byte encoding system that could enable up to 65,536 characters. The Unicode consortiums hold the world big tech companies like Microsoft, Apple, Xerox, ISO (International Standardization Organization) as a member.

The first attempt of encoding Amharic alphabet in Unicode was in 1992. The proposal submitted for Unicode Inc. by Ethiopic consists of a list of questions/ Issues. The proposal puts all Amharic characters into 96 blocks. This was achieved by splitting Amharic characters into consonantal and vocalic components as shown in Figure 3-5. Appendix IIX shows the complete set of Amharic characters arrangement in the proposal.

In this representation each character containing both consonantal and vocalic elements, are produced by rendering by treating as consonant + vowel pairs. For example Amharic character ጎ (Go) represented as U+1223(Ethiopian Consonant G) plus U+1236(Ethiopian vowel O). The character ጎ is then formed by connecting the consonant ገ and ligature of the vowel O. The problem with this proposal is that it did not consider Ethiopic (Amharic) characters as syllabic [**62**]. Syllabic is the main property of the Amharic language, lucking this characteristic makes it difficult to apply text processing and retrieval operations.

The second proposal was to address the drawback and limitation of the first proposed approach of Unicode encoding to Amharic characters. This proposal consists of block of 884 characters (Unicode U+1200 -> U+137F). Different Ethiopian professional working in high tech companies participated during the drafting process. This character set believed to cover characters needed in Ethiopic/Amharic, Tigringa, Oromiffa. Appendix IX have the full list of the current Unicode encoded Amharic/Ethiopic letters.

In the current Unicode encoded characters each cell in codespace range U+1200-> 137F represents a conceptual syllable. In this encoding level ligature are not considered. But, in case it is required like European within the Ethiopian script there may be provided as the features of particular fonts, but they are not represented as the primary character encoding.

A few Ethiopian consonants have labialized ("W") forms that are traditionally allotted their own consonant group in the syllable matrix, though only a subset

of the possible voweled forms are realized. These derivative syllables are encoded after the main alphabet, currently in the range U+1320 -> U+1347. Since the standard vowel series includes both "A" and "WA".

Amharic alphabets are encoded with the order of that they appear in traditional alphabetical table, which starts with ሀ (Amharic word HAE). The word separator in Amharic is word separator is U+1361 Ethiopian Word Space (፡), but in modern usage a plain white word space is becoming common. A separate character U+1360 ETHIOPIAN SPACE has been provided for the later usage, so that its width may be set equal to that of the traditional word space if desired

3.1.2. Keyboard Inputs

The Amharic script has more than 300 characters, yet the keyboard inputs consist of 44+8 keys. This keyboard inputs codes are expected to be passing entries that are resolved into syllabic characters before they enter stored text. Ethiopian scripts have often multiple letters corresponding to the same Latin letter. Therefore the input of these characters is done by applying certain techniques, such as swathing CAPS lock, using SHIFT button etc.

3.2. Amharic Word Formation

Amharic word can be formed by proper arrangement of phonemes of the language into meaningful unit. These units are the smallest meaningful component called 'Morpheme' (word). Morpheme is the smallest unit of the word in the language [**17**].

There are two types of morphemes. The first types are root words that can give meaning by themselves. These groups cannot further divided into smallest related meaningful words.

E.g. ተማሪ (Student)

The second Morphemes are those did not give meaning by themselves. However these morphemes can change the meaning of the independent morphemes to

form a new related word while added on independent morphemes. We can call them dependent morphemes. These morphemes cannot stand by themselves and give meaning, but have the ability to inflect the root word into different related words.

E.g. አስ- ፡ when added to the word ተማሪ gives meaning mean teacher

አስ-ተማሪ= አስተማሪ (teacher)

This behavior of word formation in the language is the major reason for morphological nature of the Amharic. In this research during experimentation we found that any averagely used Amharic root word can form about 800 other words that are related to it (root word). Below are the detail of how Amharic word formation.

3.3. Amharic Morphology

Amharic word properties are based on this kind of inflections. That's the main reason of Amharic morphological nature. There are different forms of inflections of a stem (independent morpheme) to other forms [**17**].

There are words in Amharic that are noun innately. The root morphemes that are noun are መሬት፣ ቤት፣ ሰው፣ are considered as a root noun. But there are also dependent morphemes that are noun word forming. Below are the instances of these dependent morpheme types of noun formation.

Noun forming dependent morpheme -ነት

This dependent morpheme is forming noun words from other nouns or different word parts. The Table 3-1 shows examples of noun formation by using the dependent morpheme -ነት

-ነት(dependent	

morpheme)			
ደግ	kind	ደግነት	kindness
ልጅ	boy	ልጅነት	youth
ግንበኛ	builder	ግንበኛነት	
ወንድ	man	ወንድነት	masculine

Table 3-1: example of noun forming morpheme -ነት

Name forming morpheme,-ኛኛ

New words formed this way are going to indicate characters and deeds. The Table 3-2 shows examples of nous that are formed due to inflections of the other words in the language with dependent morpheme -ኛኛ

እግር(-ኛኛ)	foot	እግረኛ	pedestrian
መንገድ(-ኛኛ)	road	መንገደኛ	passenger
በር(-ኛኛ)	door	በረኛ	Goal keeper
ፈረስ(-ኛኛ)	knight	ፈረሰኛ	Horseman

Table 3-2: Examples of names formed by using dependent morpheme -ኛኛ

Noun forming Morpheme -ኧት

ሹም(-ኧት)	officer	ሹመት	Promotion
ክብር(-ኧት)	prestige	ክብረት	respect
እውቅ(-ኧት)	famous	እውቀት	knowledge
ብስል(-ኧት)	wise	ብስለት	wisdom
ድርቅ(-ኧት)	drought	ድርቀት	constipation
ብልጥ(-ኧት)	cunning	ብልጠት	
ፍጥን(-ኧት)	speed	ፍጥነት	speedy

Table 3-3: examples of nous formed by name forming morpheme -ኧት

ውርደት፣ ንቀት፣ ጭነት፣ ቁመት፣ ጥረት even this terms follow the same type of inflection of forming a noun, further dividing the terms did not produce independent

morpheme. Therefore they are going to be considered as a root term (independent morpheme) that is going to be inflected further.

Noun forming Morpheme -ኤ

This morpheme is added on specific nouns and forms other noun/name. Table 3-4 shows examples of new name formed by using name forming morpheme morpheme -ኤ

	palace		Place of Birth
መንዝ(-ኤ)	Menze	መንዜ	A man/woman from Menze
ጎንደር(-ኤ)	Gondar	ጎንደሬ	A man/woman from Godar
ከተማ(-ኤ)	Urban	ከተማዬ	People living in city
ገጠሬ(-ኤ)	rural	ገጠሬ	Countryman

Table 3-4: examples of nouns formed by name forming morpheme -ኤ

ግሳጼ፣ ውዳሴ፣ ፍጻሜ they follow the same form of inflection even though there is no independent morpheme if we try to further decompose.

Noun forming morpheme -ኦ

The same way this dependent morpheme forms a noun from other noun. Table 3-5 shows some examples of names formed form other names with the dependent morpheme.

ጥርስ(-ኦ)	teeth	ጥርሶ	
ንፍጥ(-ኦ)		ንፍጦ	
ሞኝ(-ኦ)	full	ሞኟ	

Table 3-5: examples of nouns formed by name forming morpheme -ኦ

Noun forming morpheme -እና

The dependent morpheme -እና can form a new noun by adding at the end of the word.

ሽምግል-(-እና)	old	ሽምግልና	aged
ቁንጅ-(-እና)	beauty	ቁንጅና	beautif

			ul

Table 3-6: examples of nous formed by adding dependent morpheme -እና

Noun forming morpheme (-ኣት)

These words in Table 3-7 are formed by using dependent morpheme -ኣት at the end of the word.

ጥፍ-(-ኣት)	ጥፋት
ጥም (-ኣት)	ጥማት
ግርጥ-(-ኣት)	ግርጣት

Table 3-7: new words formed by adding dependent morpheme -ኣት

Noun forming morpheme (-ኣሽ)

ስርቅ-(-ኣሽ)-› ስርቆሽ | ቅብብል-(-ኣሽ)-›ቅብብሎሽ

Other noun forming morphemes

- ችሎታ(ችል-ኣታ)
- ውጤት(ውጥ-ኤት)
- ንዴት(ንድ-ኤት)
- ምረቃ(ምረቅ-ኣ)
- ልመና(ልመን-ኣ)
- ብሶት(ብሶ-ኣት)
- ዝምታ(ዝም-ታ)፤ ጸጥታ(ጸጥ-ታ)፤ ትዝታ(ትዝ-ታ)፤ ደስታ(ደስ-ታ)፤ ከፍታ(ከፍ-ታ)
- ግፊት(ግፍ-ኢት)፤ ንፊት(ንፍ-ኢት)፤
- ሰባኪ(ሰባክ-ኢ)፤ ፈላጊ(ፈላግ-ኢ)፤ ጭማቂ(ጭማቅ-ኢ)
- ኢትዮጲያዊ(ኢትዮጵያ-ኣዊ)፤
- መባረኪያ(መባረክ-ኢያ)፤
- እንግሊዘኛ(እንግሊዝ-ኛ)፤

There are also morphemes which are added at the beginning of the word that forms a noun. These morphemes are usually added on the verb.

- ✓ መለመን(መ-ለመን)
- ✓ መግደል(መ-ግደል)
- ✓ መሄድ(መ-ሄድ)

Numerical inflection of Amharic Language

Noun, verbs, adjectives or other parts of Amharic can be inflected into number, gender, and other forms. Noun inflections into numbers are either countable or uncountable nouns. Uncountable nouns do not show the exact number of the inflected noun. In Amharic there is no indicator for singular nouns that they are singular.

Plural words in Amharic are indicated in to two ways; one is adding dependent affix morphemes the other technique is to repeat the noun itself.

E.g. of noun numerical inflections with affixes

- ✓ ላም-› ላም(-ኦች)-› ላሞች
- ✓ ድመት-›ድመት(-ኦች)-› ድመቶች
- ✓ በሬ-› በሬ(-ዎች)-› በሬዎች
- ✓ አንበሳ-› አንበሳ(-ዎች)-› አንበሳዎች
- ✓ ቋንጣ-›ቋንጣ(-ዎች)-› ቋንጣዎች

E.g. of numerical inflection of nouns with repetition

- ✓ ጌጥ-› ጌጥ-ኣ-ጌጥ (ጌጣጌጥ)
- ✓ ጥሬ-› ጥሬ-ኣ-ጥሬ (ጥራጥሬ)
- ✓ ጨርቅ-› ጨርቅ-ኣ-ጨርቅ (ጨርቃጨርቅ)

Noun inflection to plural for those words who finish with the 6th symbol (‘Sadis’) of Amharic alphabet arrangement add the word (-ኦች) but the other nouns which finish other than the ‘sadis’ sound takes suffixes (-ዎች) to make the nouns plural.

Gender inflection of Nouns in Amharic words

The other behavior is noun gender inflection have two ways. There are two types of gender in Amharic grammar. Amharic has two types of genders; grammatical and natural gender.

E.g. of Grammatical Genders

- ✓ በግ በግ-ኢት-ኡ (በጊቱ)
- ✓ ጦጣ ጦጣ-ኢት (ጦጢት)

Some other nouns are indicated in the sentence arrangement. For instance the gender indicator might me the verb adding affix the used to indicate the gender of the noun in the sentence.

- ✓ ልጅ መጣ
- ✓ ልጅ መጣች the added (-ች) sound at the last of the verb shows the gender of the noun in the sentence. Otherwise the same noun can be used for both masculine and feminine in the sentence.

The other type of genders in Amharic is natural gender. These kinds of nouns are either male or female by nature. No grammatical arrangement to change the gender type of the nouns.

E.g. of natural genders

- ✓ በሬ - ላም
- ✓ አባት - እናት
- ✓ ወንድም - እህት
- ✓ አጎት - አክስት

Objective inflection of Amharic nouns

The other type of inflections in Amharic language is objective noun inflection, adjective noun inflection, and possession noun inflections.

ስም(ባለቤት)	ተሳቢ	ዘርፍ	ዘርፍ(ከኛ መደብ ሌላ)		
ልጅ	ልጅ-ን(ልጅ-ን)	ልጇ(ልጅ-ኤ)	ልጅህ(ልጅ-ህ)	ልጅሽ	ልጃችን
መልክ	መልክን(መልክ-ን)	መልኬ(መልክ-ኤ)	መልክህ(መልክ-ህ)	መልክሽ	መልካችን
ሙያ	ሙያን(ሙያ-ን)	ሙያዬ(ሙያ-ኤ)	ሙያህ(ሙያ-ህ)	ሙያሽ	ሙያችን

Table 3-8: objectives inflection that shows 'muya'

Inflection of Amharic Verbs

Verbs in Amharic are found at the last of the sentence. The other behavior of verbs in Amharic is they take affixes that indicate the subject of the sentence.

E.g.. ሚስቴ ዶሮ ወጥ ሰራ-ች (my wife prepared dorowet). The affix at the end of the verb indicates the subject of the word ሚስቴ.

The plural inflection of Amharic verbs depends on the tense and degree of person of the verb. That means there is a different plural inflection for the first person, second person, or third person. Beside that the plural inflection should not be applied randomly on the sentence rather it should be hand in hand with the noun in the sentence. The verb has indicator the gender and person of the word.

The general inflection of Amharic words of verbs are shown in the below table2. Seeing the table the plural inflections are of three types -ን -ኣቸው -ኣችሁ for the different class of person for each.

person		Singular Indicator	Plural indicator
1st person		-ኝ	-ን
2nd person	Masculine	-ህ	-ኣችሁ
	Feminine	ሽ	
3rd Person	Masculine	-ው	-ኣቸው
	Feminine	-ኣት	

Table 3-9: inflection of Amharic verbs

- ✓ ሀሳቤን አብራራሁኝ(-ኝ)
- ✓ ሀሳባችንን አብራራን(ን)
- ✓ ሀሳብህን አብራራህ(-ህ)/ ሀሳብሽን አብራራሽ(-ሽ)
- ✓ ሀሳባችሁን አብራራችሁ(-ኣችሁ)
- ✓ ሰውየው ሀሳቡን አብራራው(-ው)
- ✓ ሰውየው ሀሳቦቹን አብራራቸው(-ኣቸው)

E.g.. ወደደ-› ወደደ-ኣቸው(ወደዳቸው)

ወደደች-› ወደደ-ችን(ወደደችን)

Beside this verbs can be inflected to show the tense of the sentence. Verbs can show the present, past and future tense by their inflections.

E.g.. ሹፈሩ ጤፍ ገዘት-አ ነበር/ ሹፈሩ ጤፍ ገዘት-አ -ኣል shows the sentence is at the past tense from the inflection of the verb.

Adverb forming nouns in Amharic words

Adverbs in Amharic are very few, yet the role of adverb in the sentence can be played by other words.

E.g. ሌባው *በፍጥነት* ተደበቀ፡፡

From this sentence the word *በፍጥነት* is assisting the word ተደበቀ so called adverb. Adverbs can also be formed from inflection of verbs. The main morpheme that form adverb is -ኛ from other words. Some examples of adverbs are …

-ቶሎ - ገና -ምንኛ - ክፉኛ

Properties of Preposition inflections

In Amharic preposition neither inflects to other forms nor have meaning by them. Prepositions are meaningful whenever they are inserted in the sentence as part of noun or verb. Some of the examples of preposition in Amharic are ከ, ስለ, ለ and እንደ

E.g., - አባት ከሸራ ወደ ቤቱ ገባ

- ✓ ልጁ እንደ አባቱ ወፍራም ነው
- ✓ አለቃዬ ለስራ ውጭ ሀገር እንደሄደ ቀረ

3.4. Challenges Related to Character Variation

The presence of these redundant characters with the same sound in the language creates problem, especially in term matching retrieval systems. Literally different word can be formed by combining the different form of the same sound character [**52**].

This challenges come due to there are multiple alphabet in the language to represent a single phoneme. For instance the phoneme Ethiopic *HA* represented with Amharic symbols ሀ ሃ ሐ ሓ ኀ ኻ ጎ and ጋ. All this characters represented the same sound. Any time the sound *HA* appears it might take any of the eight characters. For example the following word '*hayil*' means power in Amharic and can be represented in all the ways phonetically.

- ✓ ሀይል - ሃይል -ሐይል -ሓይል - ኀይል -ኻይል - ጎይል -ጋይል

The previous research attempts try to apply normalization for those words with characters by substituting with just of the characters. This approach might short of showing the trending in the scripting behavior of the linguistic of Amharic.

E.g.. In the Table 3-10 The Amharic letters represent similar phonemes. But that doesn't mean it's correct to use interchangeably in the language.

Phoneme	Similar phonemes			
HE	ሄ	ሔ		
E'	እ	ዕ		
A'	አ	ዐ		
He'	ሄ	ሔ	ጔ	ኼ

Table 3-10: different characters for similar phonemes

Take as example the following words, which means God. እግዚአብሄር / እግዚአብሔር are the only right combination from all the possible combinations that create the word with phonemes in table 1. The following are not trending in the language in ዕግዚዓብኄር/ዕግዚዓብኔር

Even there is trending of word formation in Amharic words. For instance it's uncommon to get a word ዐገር than አገር which mean 'nation'. If linguistically technique were use it might be checked the regular type of representation of words in the language while indexing. This research applies the technique of applying for indexing.

In conclusion it is not possible to describe all the properties of Amharic word inflections. The one thing we can see from the above examples is that Amharic

words are highly morphological. Simple statistical preprocessing of text cannot handle the problem of linguistic ambiguity and linguistic variation.

The statistical approach of Amharic IR designing usually ignores or don't give enough attention to resolve the issues in section 3.2.

To handle word variation challenges of the Amharic language the statistical method apply stemming. But, it does not yield enough result for morphological behavior of the language. Section 4.2 shows the disadvantages and limitations of stemming on Amharic IR design.

Judith [**63**] mentioned that morphologically rich languages could benefit greatly from the linguistic retrieval system. Amharic is highly morphological language; it will have many linguistic and linguistic variations, trying to avoid NLP approach while designing corporate level Amharic IR system will be more likely less effective (failed) system. NLP should be given enough attention to apply it to IR. Specially trying to develop a large scale Amharic IR system might suffer a lot if linguistically approach is not considered due to the nature of the language.

This language related characteristics of Amharic make it challenging to implement statistical preprocessing in designing Amharic IR. Since Amharic is a different language in its nature, it needs IR designing approaches that address the syllabic and morphological characteristics of the language. This is why language related IR models are necessary for Amharic. This paper designed linguistic preprocessor and LM-IR using NLP technique, which is more suitable in preprocessing of the words.

In the next chapters, the detail of the designing and experimentations are covered in detail.

CHAPTER FOUR DESIGN OF LM AMHARIC IR

Many of today's IR system design approach do not consider the role of NLP/computational linguistics while designing the IR system. The researchers usually fail to incorporate the contribution of NLP for IR on their study [**25**].

The local Amharic IR design approaches have also been following the same path. They give little attention to the linguistic features of Amharic language. This is because those research approaches started by selecting a model and then try to implement it on the language. The approach they followed tried to test different models on Amharic, models that are developed for English [**13**,**14**,**15**,**12**,**21**].

But on this research showed the opposite way is more effective, starting from the language then to the model. Before designing IR system, first it needs a study on the language to gain a deep understanding of the linguistic behavior. After that it is possible to implement appropriate model on that specific language. The reason to give case for model selection is because most of the time one model that is suitable for one language might not yield satisfactory result on other language due to linguistic differences of languages. In this way by starting with detail linguistic study on the language, it is possible to choose the right model, if there is no one, it's better to design one to suit for the

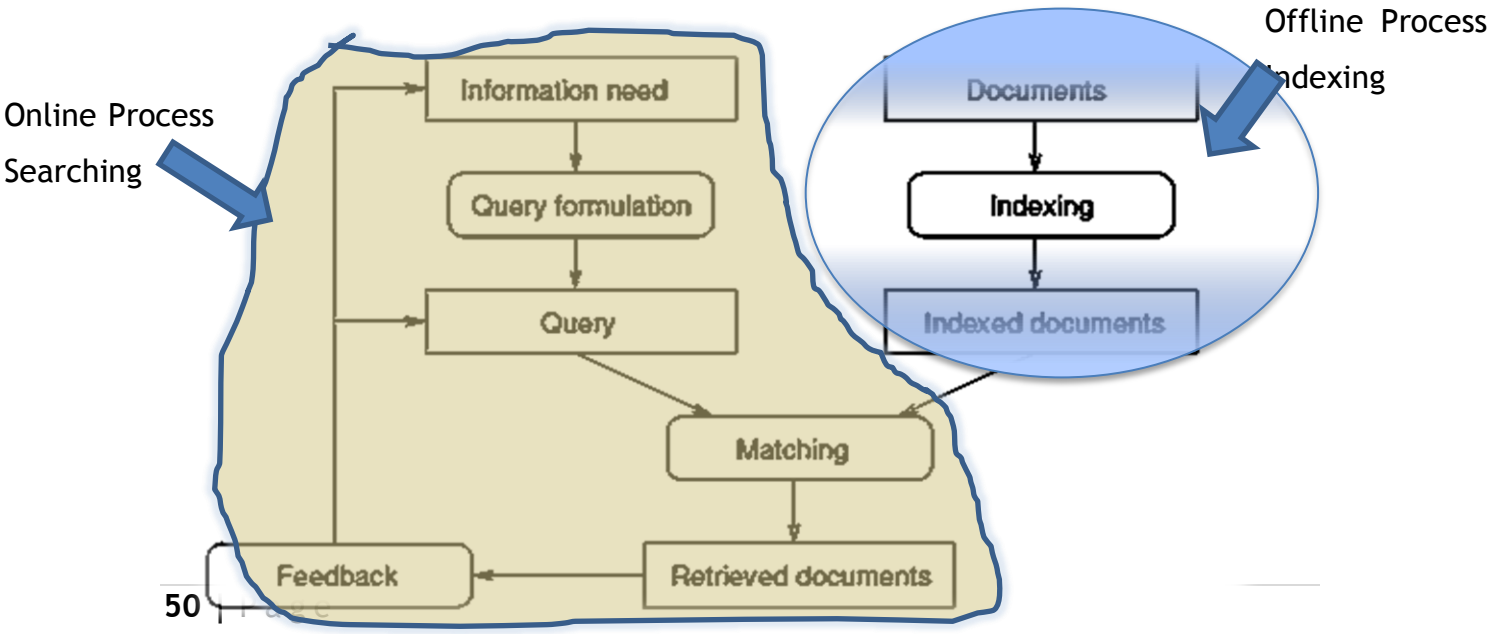


Figure 4-1: General IR Model

linguistic behavior.

Any IR system design, whether it's linguistic or statistical have the same subparts, indexing and matching. These two subsystems of IR will be always inseparable. If something has to be searched, it should be first indexed. Indexing is the offline process of preparing documents for effective searching process. searching is an online process that complement of indexing by matching users' information need represented in users' query with indexed information. Figure4-1 shows the general IR system model.

The main difference of statistical and linguistic IR systems designs is the approach and techniques applied while implementing these sub systems. Nonlinguistic approach of IR system design follows purely statistical techniques for both indexing and searching. But the linguistic approach of IR aimed to apply NLP technique for implementations of indexing, weighting, matching and/or ranking of the IR system while designing.

Nonlinguistic method of Indexing is based on the statistical behavior of terms in the document, such as the count of words in the document or presence/absence in the case of Boolean model. The content bearing ness (importance) of the term in the document is determined by the count of terms in the document. This model of indexing doesn't attempt to guess the semantics of the terms in the language or in the corpus. It ignores the relationship of terms with each other. The approach is incapable of calculating linguistically significance of terms or relative meaning/ information content of the term in the document that contain it.

The idea of linguistic technique of IR system design is to resolve the drawback of statistical technique of IR system design. NLP motivated IR designing technique tries to find the meaning of terms and their semantic relevance with in the collection or in the language, rather than relying the counting of terms in the document as the only factor. If it is able to implement linguistic has ability to semantically represent documents than statistical approach.

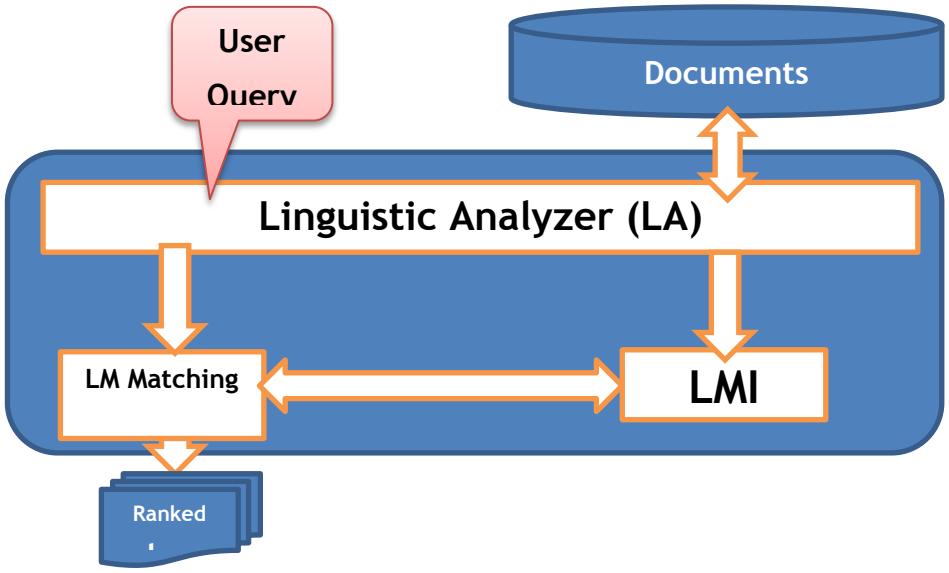


Figure 4-2: LM-IR system design

NLP motivated IR system design will resolve the problem of linguistic ambiguity and linguistic variation. Amharic as heavily morphological language could benefit from linguistically approach of IR system design. Judith [**63**] mentioned that morphologically rich languages could benefit exponentially from the linguistic retrieval system. Since Amharic is one of the most morphological languages whose characters are syllabic. Applying LM IR on the language can open a new dimension of IR for the language.

Previously, applying NLP techniques for IR system designs thought to be avoided for their performance issues during 80s and 90s, because of this there is no full-fledged NLP motivated IR model, as it does in statistical approaches.

On this research an effort has made to develop NLP model that syntactically analyzes Amharic terms before indexing to determine the semantics of the words in the language. This model also provides a linguistic way of Indexing and matching. The basic model shown on Figure 4-2 shows the proposed model for linguistic IR system design.

As the model in Figure 4-2 depicts the statistical preprocessing replacement is the Linguistic Analyzer (LA). The LA substitutes the preprocessor of the

statistical approach. It analyzes the document terms and query terms before indexing and matching operations. The goal of applying linguistic technique is to be able semantically determine the meaning contained within the term that used to index the document (information).

The Figure 4-3 and Figure 4-4 shows the detail model of statistical vs. newly developed linguistic IR models respectively. Closely looking and comparing the two models we can realize that the major difference of the two models is on the preprocessing part. But it also uses modified type of indexing structure, weighting technique and matching approach.

The reason this research focus on preprocessing, it is because most of the challenges/limitation in Amharic IR system design is due to meaning degradation during preprocessing.

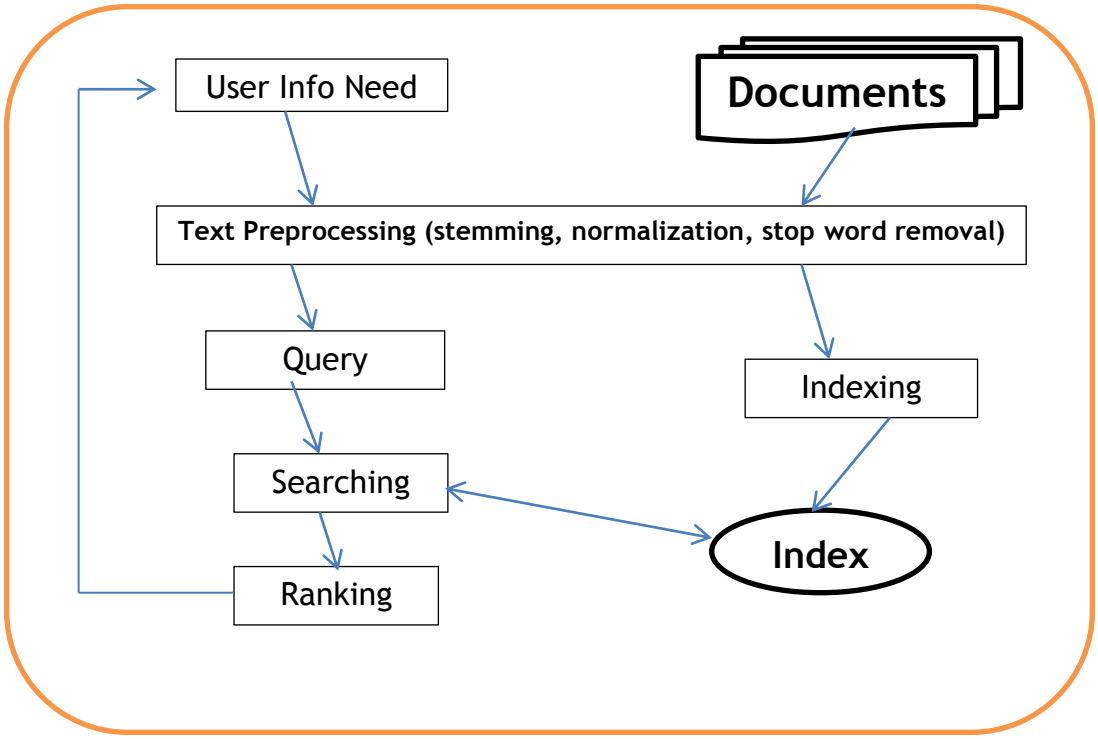


Figure 4-3: statistical IR structure

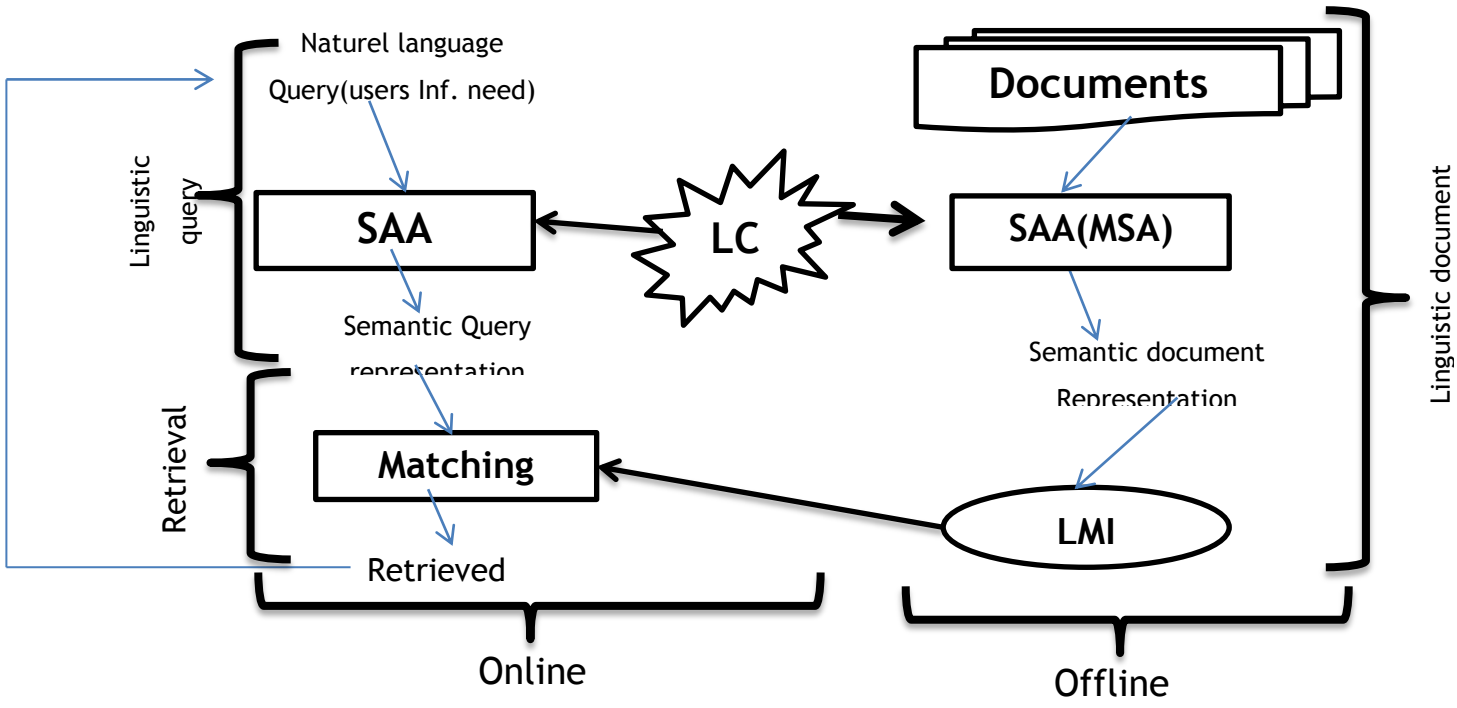


Figure 4-4: LM IR detail diagram

Preprocessing technique comes before indexing of a term. The goal of preprocessing is to increase the performance of the retrieval system by removing non information content words and grouping related words with one representative. Documents after preprocessing should have a list of index terms that are representative of the inflected the whole document.

Here is shown the comparison of the two preprocessing approaches, the statistical vs. linguistic approaches on Amharic. The statistical approach adopt the well know technique of text preprocessing like stemming, normalization, and stop word removal that have been used effectively on English. On the other hand there is no fully defined and recommended linguistic preprocessing approach; therefore syntactical preprocessing technique suitable for Amharic is developed after testing different approaches. The linguistic approach of preprocessing proposed because of the limitation of statistical approach on Amharic language. Below it is showed the limitations of statistical approach of preprocessing on Amharic language.

4.1. Limitation of Statistical Preprocessing

Most researches on Amharic IR usually ignore to incorporating linguistic rules of the language. Those researchers do not give enough attention to the linguistic part while designing Amharic IR. One of these indicators is the effort of implementing of preprocessing. Those researches try to implement well known models of IR preprocessing like stemming, stop word removal, and normalization without realizing the it's adverse effect of the languages

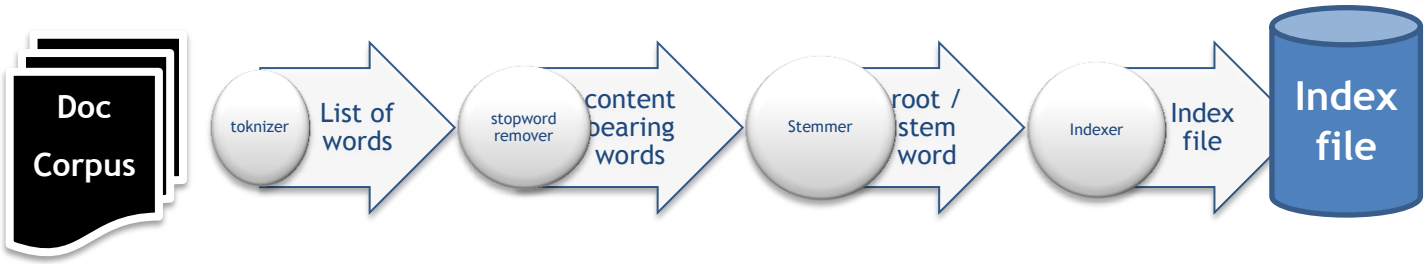


Figure 4-5: Statstical Preprocessing

retrieval system.

Figure 4-5 shows the statistical preprocessing. The definition of each step and its impact on the language preprocessing is stated below.

4.1.1. Statistical Tokenization

The first step of preprocessing in statistical preprocessing is tokenization. It is the process of changing a document into sequence of discrete tokens (words). This step is the same for both linguistic and statistical approach as long as the retrieval system is based on the word level. Simplest approach of tokenization is to ignore all numbers and punctuation and use only unbroken strings of alphabetic characters as tokens.

4.1.2. Stop word Removal (Statistical)

By definition stop words are terms that do not have meaning related to the document. That means the presence or absence of these terms in the document does not have a change in the overall meaning of the document. The intention of stop word removal is to increase performance without affecting effectiveness due to removal of stop words.

In previous attempts stop word removal was done by collecting and categorizing Amharic words as stop word list. The recent attempt is by Ammanuel [**15**], his full list of stop words shown on Appendix II. To understand the impact of stop word removal on Amharic IR retrieval let's take one word from the list and see what will happen when it is removed.

The word **ገና** is considered as non-content bearing term it is found in stop word lists. A document having the following sentences which talk about Christmas uses the term **ገና** which considered as stop word in the Amharic words

Example 4-1: The impact of stopword removal on Amharic

ገና ከመድረሱ በፊት ስራችሁን ጨርሱ..... ምንያቱም ልጆችን ለገና በአል ሸርሸር መውሰድ አለብን።። .....ምንግዜም ቢሆን በገና እለት ህጻናቱ ድራማ መስራት ስለሚፈልጉ ሁኔታዎችን አመቻቹላቸው።።..... ለነገሩ በገናም ሆነ በፋሲካ ይህ አይነት ልምድ አላቸው።።

The terms **በገና**, **ገና**, **ለገና**, and **በገናም** are all inflection of the word **ገና**, so they have the same root. If we consider this term as stop word and remove it from the list of tokens, the central meaning of the document will be lost. We can clearly see from Example 4-1 the word ገና is a content bearing. Removing it might lose the center of focus or the retrieval system; consequently we will not be retrieving the document.

Considering the list of words as if not content will bearing the effectiveness of the system accordingly. The linguistic approach might provide a remedy for this kind of problem.

4.1.3. Stemming (Statistical)

Purpose of stemming is to group linguistically interrelated terms into a single term that can semantically represent all the other derivations. The statistical approach involves the process of identifying prefixes; infix and suffixes in order to get the root (stem) word from the inflections.

Stemming technique reported to be effective on English language IR, but not always in Amharic. To see the impact of stemming on Amharic with example

Example 4-2: The impact of stemming on Amharic

The Amharic term መጣች(She came) can be stemmed to መጣ(he came) by removing the suffix -ች. But the word መጣ(third person masculine) and መጣች(third person feminine) they are not equivalent. Therefore morpheme -ች is not just to be removed. It holds extra information that might lose if it is truncated by stemming procedure.

Example 4-3: The impact of stemming on multiple-inflected Amharic word

Whenever the word is inflection multiple times the information lost due to stemming is a lot worsen. Let's consider a word "አላበራችባቸውም" which means "She did not light on them".

አላ-በራ-ች-ባቸው-ም (this word contain information related to tens gender action and number) after stemmed results በራ (means "Lighted"), which are not equivalent.

From Example 4-2 and Example 4-3 we could see meaning loss due to stemming. Stemming operation on Amharic document can cause a loose of semantic meaning of the document. The inflections in Amharic hold information related to gender, number, tense and possession. It needs to apply linguistically technique not to lose the meaning related to the word.

The other disadvantages of stemming come from the lists of morphemes that are considered to be prefix, suffix or infix. The stemmer Algorithm [**14**]

truncates all the terms that start with the prefix list and ends with the suffix list. This might not be always the case.

For instance some researches puts a list of possible prefixes and suffixes, appendix VI shows the full list. In addition there infixes that inflect the meaning of the word. Let's take one word from the list and see the impact of applying prefix removal.

Example 4-4: Non representative list of affixes

እስከ-መጣ(which means 'until I come'): correct

እስከ-ንድር(not meaningful): incorrect

As the Example 4-4 shows እስከ- considered to be a prefix in the word inflection መጣ by forming a tem እስከ-መጣ(which means 'until I come') . But we can see that all words that starts with እስከ- are not inflected the same way. እስከንድር is not, because after removing prefix እስከ- the remaining -ንድር is not meaningful morpheme, even it is not dependent morpheme.

In the same way there are many words that begin with the prefix list and ends with the suffix list but not linguistically inflections. Single root term

The other issue of non-representativeness is related with lack of stating the entire suffix and prefix lists in the document language. the study experimentation able to show an averagely used Amharic word can inflect as many as 800 see (Table 4-2) times to form different words. Therefore the list of prefix and suffixes stated not more than 100. The other challenge of trying to list all possible prefix and suffix lists is related to false stemming. That is, stemming words that should not linguistic, as stated in non-representativeness on Example 4-4.

4.2. LM Preprocessing [Proposed]

After corpus preparation, Linguistic analysis of NLP performed at preprocessing and matching phase. The preprocessing tries to grasp the concept of inflections of the word by analyzing the given term with the corpus and getting its

inflections. This technique does not involve any traditional preprocessing techniques.

To overcome the challenge of semantic degradation of Amharic words due to statistical text processing, on this research we shown a corpus based linguistic Morph Syntactic Analyzer (MSA).

MSA is a corpus based document/query term preprocessor Algorithm that determines the semantics (meaning) of terms in the document/query. The corpus in LA supposes to have all possible Amharic words without any statistical preprocessing. Since Amharic is highly morphological rich language there is a lot of information loose due to stemming, normalization and stop word removal. By using Surface (morphological) syntactic analysis to generate term variant patterns from the application of morph syntactic transformation into word parts which would be possible to save terms meaning degradation that might cause by stemming.

The basic goal of the Linguistic approach of word preprocessing is to get the semantic representatives for common word groups. The output of this processing is a term that represents the true meaning of the word in the language. After preprocessed with linguistic approach those words are semantic representatives of the document they in.

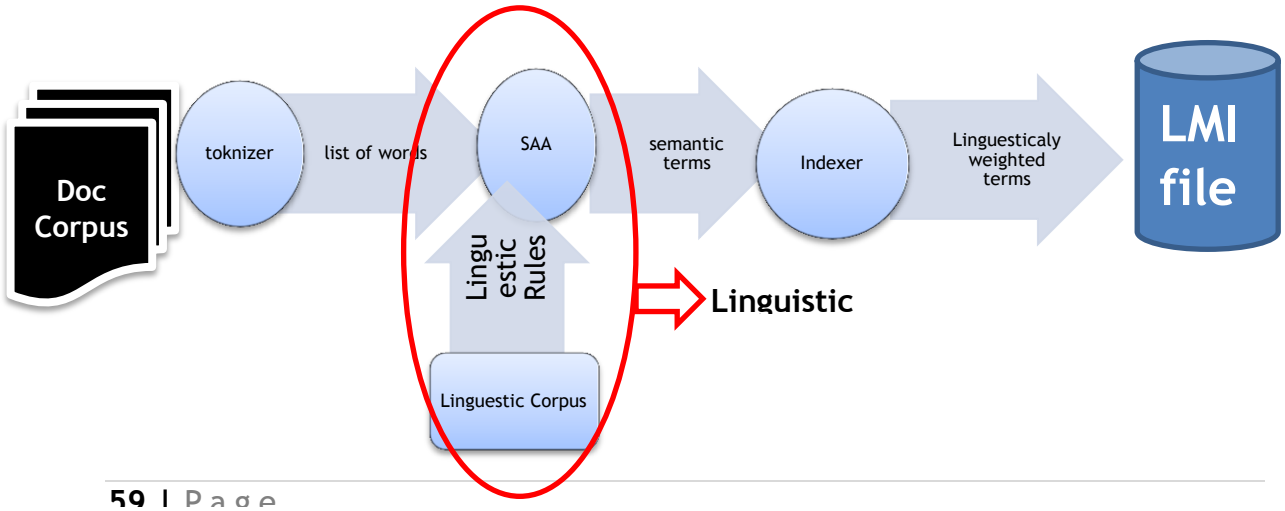


Figure 4-6: Linguistical Preprocessing and Indexing

The core idea this research is to design the linguistic preprocessing analyzer using the syllabic nature of Amharic writing system. The preprocessing task should be done at word level, according to the linguistic behavior of the language. Some of these behaviors are found in sections 3.1.3 and 3.2. The step followed to design this system is done by applying the following steps.

4.2.1. Tokenization [for LM preprocessing]

Tokenization follow the same procedure as it is in the statistical approach as long as the indexing process works at word level. The goal of tokenization is to produce a list of linguistically correct list of words from the file in the documents.

During tokenization there are a number of steps for data collection and string clearing. The best approaches of preprocessing to get words that are suitable for Linguistic analysis using morph syntactic are indicated below.

4.2.1.1. Proper file acquiring

This research tried to implement automatic document collection mechanism. The Algorithm iterates though all hierarchies in the root directory to collect files the necessary for operation. To come up with this, the following steps should be performed.

- ✓ **Step1:** Collect filenames of any text file in the root directory: Text files in the root directory should be checked and the file names should be stored in a list. This operation shown in the **Error! Reference source not found.** and its implementation of python is shown in appendix V.

```
FileNameList=[]
Dir=Root_Directrory
For all Files in Dir
      If filename  ends with ".txt"
            Append filename to FileNameList
      Dir= Child_Dir
Return FileNameList
```

Algorithm 4-1: Opening all text files in the root directory

- ✓ **Output of step1:** the module at this stage returns a list of any text file in the root and sub directories with their path of location in the corpus.
- ✓ **Step 2**: since all the text files are not important to process for Amharic we should only check those files that are Unicode encode. The Algorithm 4-2**Error! Reference source not found.** below shows how to check files with Unicode encoded. If the file is not Unicode encoded it's no use to store the file name for further processing.

```
For file in FileNames
   If file encoding != 'utf-8'
            FileNames.Remove(file)
Return FileNames
```

Algorithm 4-2: filtering only Unicode encoded files only

Since there is no way to check straight whether encoding of a file to be Unicode or not in python, it takes extra effort to implement it. One of the best approaches is to check the files ASCII encoding and handle Unicode exception errors. By checking whether the file is ASCII or not, if the module raises Unicode decode error, then the file is a Unicode encoded file it is considered to be Unicode encoded, it is implemented as well in python program at appendix V.

- ✓ **Output of Step2**: is a qualified filename list; those filenames are Unicode encoded. These files are the collected corpus that are Unicode encoded plus put in order.

4.2.1.2. Text cleaning process

A documents file can contain character inscription of different language, punctuations, diagrams, illustrative pictures those are not important in text retrieval system. Therefore, text cleaning is the very crucial task of tokenization. After a file is opened it needs to be cleaned in order to find the proper tokens. The most common type of cleaning process involves removal of non-Amharic characters, punctuations and numerals.

- ✓ **Step1**: forming tokens using delimiter. On this step the Algorithm should open and every file in a collection and identify valid tokens of the language using delimiter (for instance space). This collection of tokens supposed to hold all words from the document.

  Cleaning on Amharic characters from the file can be performed multiple ways. The best way and this paper's recommendation is identifying the range of Unicode character orders of Amharic words and filter only those which are in that range. **Error! Reference source not found.** shows the cleaning procedure by using this procedure

```
AmhPunc= [list of Amharic punctuation]
For File in FileNameList
      str=open(File)
      for char in str:
            If UnicodeOrderr(char) not in range(4600—5000) or if char in
            AmhPunc:
            Else:
                  Continue
      word=str.split(“ “)
      append ((word)) to AllTokens
Return AllTokens
```

Algorithm 4-3: Filtering out non Amharic characters

The output of this step is a list of tokens that have meaning related to the document. The proposed approach of string cleaning can simplify the process of cleaning other language alphabets. The Amharic related punctuation can be cleared separately according to their behavior.

4.2.2. Terms significance

Term significance provided remedy for the challenges related with stop word removal on Amharic documents shown in section 4.1.2. The main limitation of

stop word removal technique of statistical approach is because it applies by predetermining of list of words as stop words based on their frequency. If a word that appear most documents is considered as stop word.

Stop words believed to be non-content bearing terms in the document. In Amharic the only non-content bearing terms are adverbs [**17**]. In Amharic to identify these stopword list relying on frequency of terms in the overall terms did not yield the expected result. However, from Amharic linguistic behaviors we see that stop words are those terms is that they could not change to other inflect. The words like ከ, ስለ, ለ and እንደ did not inflect to other related terms, and these terms are stopwords (Amharic adverbs). From the above linguistic behavior we can conclude if a word is inflected in the document, then there is some meaning related with it. In other words if the word have no a capacity of inflecting then it is a stop word.

To show this with example let's consider Example 4-1, which show the limitation of statistical stop word removal and see how it provide a solution by applying inflection to identify the significance of the term in the document. The word **ገና** has two meaning, one mean "until" which can be considered as stopword, the other meaning is "Christmas". "Christmas" is content bearing term than "until". We can identify which one is Christmas and which one is saying "until" since the term **ገና**, that mean "until" cannot be inflected, whereas the term **ገና** that mean "Christmas" can be inflected into words **በገና**, **ገና**, **ለገና**, and **በገናም** of terms in the same document. And definitely there is content bearing (meaning) related to that word and its inflection.

The significance calculated using the inflection concept during the weighting of a term to identify important and non-important terms in the document. To achieve it, this research introduce a new metric called inflection degree (Id) to the IDf of measuring based on experimentation on Amharic word behavior of inflection.

$$Id_t = \frac{Inf_{t,d} * Inf_{t,C}}{tf_{t,c}}$$

$where$:

$-Inf_{t,d}$: $the\ inflection\ of\ the\ term\ in\ the\ documents$

$-Inf_{t,C}$: $the\ inflection\ of\ the\ term\ in\ the\ whole\ corpus$

$-\ tf_{t,c}$: $is\ the\ collection\ term\ frequency\ in\ the\ corpus$

From the above formula we can learn that if the ratio of term inflection to its frequency determines the importance. That means if a term in Amharic to be important it has to have inflections which means different form of information representation in the overall corpus.

4.2.3. Regular expression

The linguistically variation of Amharic make it difficult to apply stemming. To resolve the limitation and challenges of stemming on statistical part, it needs detail study linguistics of Amharic. Applying prefix and suffix list for stemming would be a nightmare for large corpus where there is multiple inflection of a single word. Plus it's impossible to mention the entire prefix and suffix lists of the language.

Regular expression is shown on this research as alternative for challenges of Stemming on Amharic document. It is done on corpus based Syntactic analysis by collecting linguistically related words in a more efficient way to represent these related words with semantic vocabulary.

Before applying the regular expression on the corpus of words need to be in sorted order. Sorting of all corpus tokens deliver better grouping of related words that are of the same root for indexing than the statistical approach of suffix removal.

For instance by sorting the corpus collection of tokens it is possible to find suffix inflected relatives of the root word ሄደ' without any intervention in the list order. As indicated in the Table 4-1.

1. ሄደሃል	9. ሄዳም
2. ሄደህ	10. ሄዳችሁ
3. ሄደሻል	11. ሄድሁባቸው
4. ሄደች	12. ሄድሽ
5. ሄደና	13. ሄድን
6. ሄደዋል	14. ሄዶአል
7. ሄደው	15. ሄጃለሁ
8. ሄዱና	16. ሄጂ

Table 4-1: consecutive words that are related to the root word

Sorting all Amharic words with their Ethiopic character order this way, it's possible to get all the terms that are of the same root consecutively. This is possible because Amharic scripting is syllabic, where inflection of the word uses little structure difference (morph variation) at the beginnings of the word as shown in the Table 4-1.

The sorting operation can be performed by simple sorting command

$$sortedList = corpusList.sort()$$

But sorting could not help in collecting related terms that are caused by prefix inflection. This is because prefixes determine the order of position in the sorted list.

Prefix variation can be handled easily by formulating regular expression on top of the sorted list of linguistic corpus. Regular expression on the linguistic corpus used for getting related terms that are not found consecutively in the sorted list.

The root word '*ሥራ*' is inflected in to 50 different words using only suffixes. To consider other types of prefix inflections in Amharic, we must design a regular expression using corpus based approach.

✓ ሥራህንና	✓ ሥራቸውንና	✓ ሥሩ	✓ ሥራውንና
✓ ሥራለት	✓ ሥራችሁ	✓ ሥሩም	✓ ሥራዎቻቸው
✓ ሥራልን	✓ ሥራችሁም	✓ ሥሩባት	✓ ሥራዎቻቸውም
✓ ሥራልኝ	✓ ሥራችሁን	✓ ሥሩት	✓ ሥራዎችዋም

✓ ሥራም	✓ ሥራችሁንም	✓ ሥሩን	✓ ሥራዬን
✓ ሥራሽ	✓ ሥራችን	✓ ሥሩንም	✓ ሥራን
✓ ሥራሽም	✓ ሥራችንም	✓ ሥራ	✓ ሥራንም
✓ ሥራሽን	✓ ሥራችንና	✓ ሥራህ	✓ ሥራዋ
✓ ሥራባት	✓ ሥራችንን	✓ ሥራህም	✓ ሥራዋም
✓ ሥራቸው	✓ ሥራና	✓ ሥራህን	✓ ሥራዬንም
✓ ሥራቸውም	✓ ሥራቸውን	✓ ሥራህንም	✓ ሥራውም
✓ ሥራው	✓ ሥራቸውንም	✓ ሥራውን	✓ ሥራውና
✓ ሥራውንም	✓ ሥራዬ	✓	✓

Table 4-2: suffix inflections of the word 'ሠራ'

During experimentation we see that most frequently used Amharic words have the capability to produce about 800 inflections due to prefix and suffix resulted from a single root word.

For instance the possible regular expression that used to collect both prefix and suffix inflected words for the Table 4-2 is indicated in below

> *For the word 'ሠራ' below, we can put a regular expression of [*ሠ[ራ]*] for all the related terms.*

Even regular expressions are the best approach to analyze interrelation of words using their morph. There is a limitation of implementing it to actual programming languages. This challenge arises because of Unicode. It is impossible to compute the regular expressions on Unicode characters like those in ASCII representation. This hinders to harness powerful regular expressions for the word analysis on Amharic alphabets that are Unicode encoded.

For this reason supplemental regular expression model developed to be used for the intended purpose from scratch. This makes it the initial morph syntactic analysis very heavy. It takes 318hrs 2min 43 sec (13days 6hours 2min and 43 sec) to finish the analysis of 74,045 Amharic words to get the linguistic interrelation.

4.2.3.1. Designing the regular Expression

Develop a custom regular expression the works on top of the sorted list to analyze and determine linguistically meaning of a term in the document. The regular expression supposed to identify linguistic meaning of the terms by syntactic preprocessing.

The challenge to regular expressions is that programming language does not give capability to implement the full potential of the regular expression on Unicode characters. Since it is the only option to effectively collect semantic related terms is to develop one for the purpose that works on Ethiopic characters.

The Algorithm developed on this model is indicated on Algorithm 4-5. What it does is that. It takes a word and tries to guess the root word from the possible prefix and suffix lists. Based on the guessed root word lists of seven different words are generated to be checked in the linguistic corpus.

```
Function rooter(word)
      If word.startsWith(pref)and word.endswith(suf):
            Word=word[len(pref):len(word)-len(suf)]
      For i from 1 to 7:
            Wordlist.append(word[:len(word)-1]+unicar(word[len(word)+i]))
Return wordlist
```

Algorithm 4-5: a function to produce root term

To show this with example let's take related words to the word ሄደ "hede" from sorted lists and perform rooter Algorithm above. ሄደሁ is the first in the sorted list, the Algorithm produce the following results….

Those results can be found according to the system type we want. This technique enables to construct a multilayer indexing, which improves both precision and recall at the same time

Guessed root word ሄደሃ-ል		
Results from rooter	Linguistic corpus test	Examples
ሄደሀ	Valid	ሄደሀል
ሄደሁ	Not valid	
ሄደሂ	Not valid	
ሄደሃ	Valid	ሄደሃል
ሄደሄ	Not valid	
ሄደህ	Valid	ሄደህ፣ ሄደህ፣ ሄደህም
ሄደሆ	No valid	

Table 4-3: precision oriented RegEx analysis

The bigger index holds all inflection of the word in every degree; the inner index holds all words with specific inflection only; for instance gender or number, or other form of linguistic variation caused by inflection.

Guessed root word ሄደ-ሃል		
Results from rooter (Regular expression root)	Linguistic corpus test	Examples
ሄደ	Valid	ሄደች፣ ሄደችም፣ በሄደችየሄደው፣ የሚሄደው፣ የሚሄደውም፣ የሚሄደውን፣ የሚሄደውንም፣ የሚሄደውንና
ሄዱ	valid	ሄዱ፣ የሚሄዱ
ሄዲ	Not valid	
ሄዳ	Valid	ሄዳችሁ፣ ሄዳችሁም፣ በሄዳችሁ፣ በሄዳችሁበት
ሄዴ	Not valid	
ሄድ	Valid	ሄድን፣ ሄድንና፣ በሄድንበት፣ በሄድንባትም፣ የሚሄድንም
ሄዶ	valid	ሄዶለል፣ ሄዶለልና
ሄጀ	valid	ሄጀለሁ፣ ሄጀለሁና

Table 4-4: Recall oriented RegEx analysis

```
Function RegEx(word, Collection)
      Wordlist=rooter(word)
      For term in Collection:
            If term.match(wordlist):
            Related.append(term)
      return Related
```

Algorithm 4-6: Identifying linguistically related terms

Algorithm 4-6 uses the rooter function from Algorithm 4-5 using the developed regular expression module in the rooter function. This is what is called the semantically syntactic Analyzer the LM-IR diagram. After trying different syntactic analysis technique for Amharic word analysis that could address the

linguistic behavior, two techniques the Syntactic Analysis Algorithm should do for getting terms that are of the same linguistic root.

4.2.4. Advantages of LM Preprocessor for Amharic IR

Advantages of corpus based syntactical analyzer over statistical preprocessing techniques are in derivational morphological system. In which reduces semantic degradation due to stemming, stop word removal.

- ✓ Derivation analysis can improve the semantic meaning of the terms
- ✓ Capable of returning more information on family of related forms and words than the basic stemming Algorithms,

LM preprocessing uses syntactical analysis Algorithm based on the linguistic rule of the language to identify terms in the collection of the corpus. This LA Algorithm tries to get the semantics of the word by syntactically analyzing the words in the language. Any word from the document (query) before indexing or matching analyzed with LM preprocessor. It is the main component of the linguistically motivated IR system that used to analyze every term before indexing it to get the semantic meaning.

- ✓ This linguistic approach has also multi-layer indexing. This indexing can represent more information representative (precision oriented) or a manner of recall oriented with simple steps. If we apply Table 4-3 and Table 4-4 we will get different indexing output.
- ✓ One of the other best advantages of linguistic indexing is to not loose inflection knowledge representations. The information related with the gender, number and tense are tried to retain the linguistic indexing approach shown on the Table 4-5.

The following terms are related with ሄደሃል • ሄደሃል • ሄደሃልና *The following terms are related with ሄደሀም* • ሄደሀ	*The following terms are related with ሄዳችሁም* • ሄዳችሁ • ሄዳችሁም • በሄዳችሁ • በሄዳችሁበት

• ሄደህም *The following terms are related with* ሄደሻል ሄደሻል *The following terms are related with* ሄደች • ሄደች • ሄደችም • በሄደች *The following terms are related with* ሄደን • ሄደና • ሄደን *The following terms are related with* ሄደውም • ሄደው • ሄደውማል • ሄደውም • የሄደው • የሚሄደው • የሚሄደውም • የሚሄደውን • የሚሄደውንም • የሚሄደውንና	*The following terms are related with* ሄዱሁባቸው • ሄዱሁባቸው • በሄዱሁበትም *The following terms are related with* ሄድሽ • ሄድሽ • ሄድሽበት *The following terms are related with* ሄድንና • ሄድን • ሄድንና • በሄድንበት • በሄድንባትም • የሚሄድንም *The following terms are related with* ሄዶአል • ሄዶአል • ሄዶአልና *The following terms are related with* ሄጃለሁና • ሄጃለሁ • ሄጃለሁና *The following terms are related with* ሄደዋልና • ሄደዋል • ሄደዋልና

Table 4-5: inflection information retaining indexing

4.3. Indexing

Let's see how preprocessing affect information content of words that need to be indexed. To see the difference of statistical and linguistically preprocessing on Amharic words with example, the sentence that mean "Abebe's flower not arrive until Christmas"

The sentence "የአበበ አበባ እስከገና አይደርስም።።" with statistical preprocessing would end up looking like Figure 4-7 below. The final result of the statistical preprocessing activity gives only two words አበበ and ደረሰ ['Abebe' and 'arrive'] for indexing; those terms are less likely to talk about the event. In most of the

time statistical word processing loose content bearing terms of the document or the language during statistical preprocessing.

This is due to the nature of linguistic variation of Amharic language, statistical preprocessing heavily affects the information content of the words to be indexed.

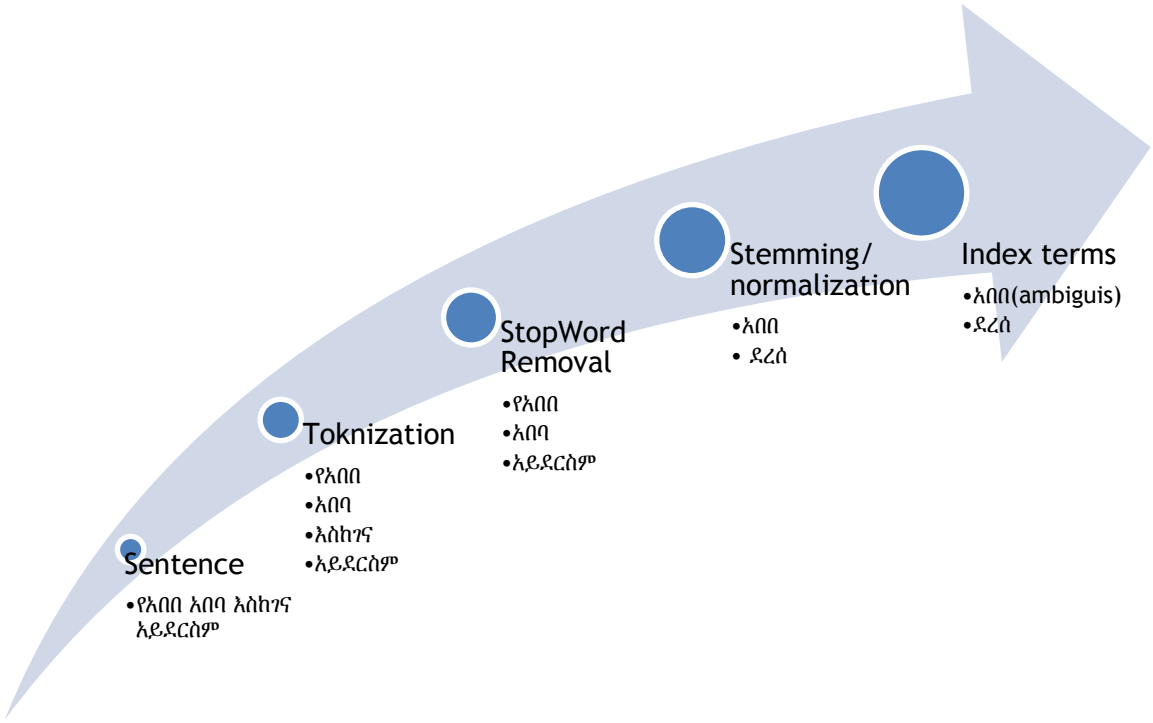


Figure 4-7: Non Linguistic text preprocessing for indexing

On the other hand preprocessing the English word will result [abebe; flower; arrive; [until]; Christmas]. This collection of index term has the whole story of the document (sentence).

This nonlinguistic preprocessing for indexing has been reported to be effective for English IR systems. This effectiveness is the result of the linguistics of the English, the root term inflection types can be predicted easily. In English morphs of the word inflections are few and predictable for most of the words, in addition exceptions are easy to handle. For instance –ed is to make past tense and similarly –s usually to make it plural or third person singular.

Applying nonlinguistic terms on these words can yield good result since the linguistic variation of English is by much less than that of Amharic

Whereas, Amharic is a different story; there is great deal of morph variation in words that are found in the language which should be addressed accordingly by studying the inflection behavior of the language. As indicated in Table 4-2 averagely used Amharic word has about 800 possible variants that caused due to inflections. Therefore it needs to come up with the way of predicting the true meaning of the term by using linguistic analysis of words with respect to the language.

Index terms in LM-IR are not just terms; they should also have meaning representation of the document. That means the indexed information is semantic meaning of the documents represented in the terms. Since more meaning is on the inflection of the word in Amharic, Inflections should be given enough attention in order to retain the information content. On this research, we tried to index not only the root word but also the concept of inflection with in the document.

The idea of conceptual index term preparation is shown in the Figure 4-8. This process of identifying index the concept using corpus based approach is called Morph syntactic Analysis (MSA).

MSA for term transformations are better adapted to the recognition of term variations than general grammar rules because they provide a framework for articulating syntactic modification with morphological changes that cause word formation.

The model below shows that how linguistic preprocessor change each words into linguistically correct index term. The analyzer uses a corpus based approach of syntactical analysis with all possible word of the language. The analysis will give index/query term that are linguistically correctly represent

the document they are in. This smallest semantically correct term will be taken as index term.

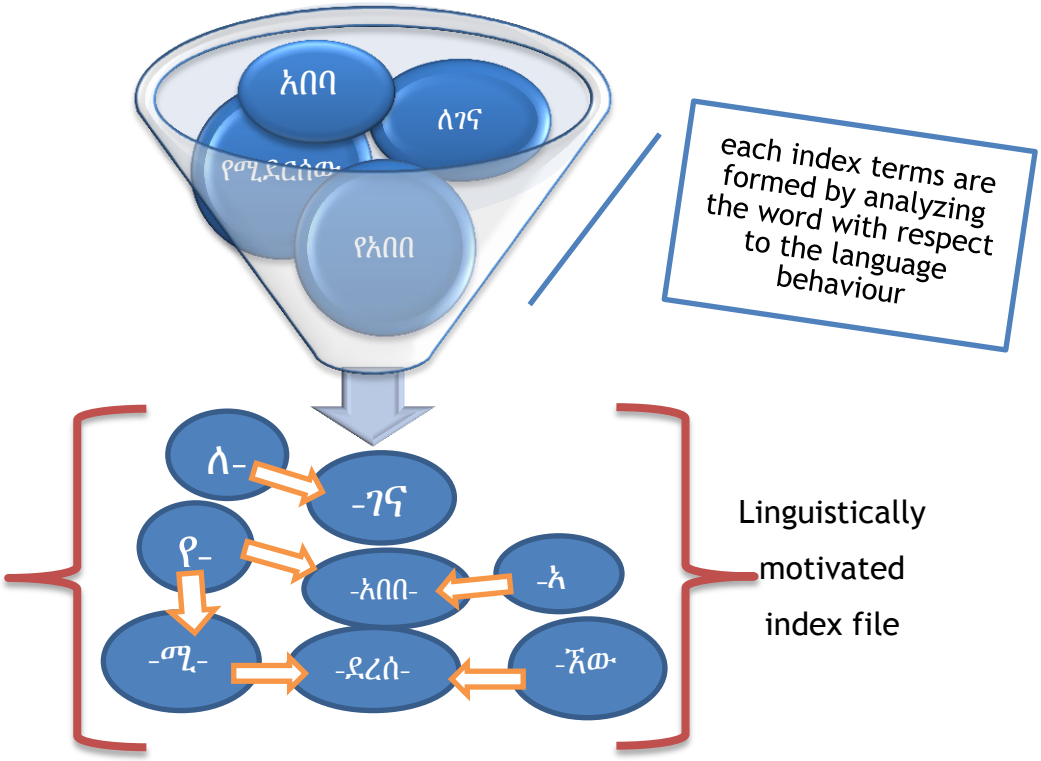


Figure 4-8: LM-IR indexing using MSA

4.3.1. LM Indexing Structure

Linguistically Motivated Index Files (LMI Files) are indicators of concepts of linguistic indexing concepts may in some times adequately express in single words. The concepts being indexed frequently have an internal structure requiring expression as so called 'pre-coordinate' terms that are linguistically well defined word units [**31**].

Index term corpus is a linguistically analyzed word where the index terms are manually marked up. It is the training and test material of the new automatic indexing method of this thesis. For indexing the analyzed parts of words used the following indexing components. The linguistic motivated index file will have the following parts.

- Conceptual Vocabulary
- LM Inverted File
- LM Weighing

4.3.1.1. Conceptual Vocabulary

It is a vocabulary file that is the conceptual representation of terms related to it. For example -ቤተ- is the conceptual vocabulary file for the words related with ቤት. These words are produced by SAA. The same query words are changed to conceptual vocabulary file

4.3.1.2. LMI file structure

The inverted index (sometimes called inverted file) is the central data structure in many information retrieval systems. The concept is that a document is assigned a list of keyword with weight associated with each keyword. And these keywords have a link to the documents containing that keyword. At its simplest, an inverted index provides a mapping between vocabulary (semantic vocabulary for this case) terms and their locations of occurrence in a text collection.

The dictionary lists the terms contained in the vocabulary V of the collection. Each term has associated with it a postings list of the positions in which it appears, consistent with the positional numbering in linguistically motivated Indexing Inverted file (LMI Inverted file) structure is a suited for the use Inverted file with minor modifications.

4.3.2. Building Inverted File

Inverted file has two major components; vocabulary file and Posting file. The vocabulary is where preprocessed unique keywords are stored. Posting file is data structure that stores the locations of these keywords in the document and frequencies. In this research sorted array method used for creating LM inverted file. Inverted file creation requires passing through several operations. First the

individual words must be changed into semantic vocabulary that is representative of the term inflection meaning (which is mentioned in preprocessing part section 4.2.)

Conceptual vocabulary	Pointer to Posting file
ሆነመ	122
ሆነሰ	123
ሆነሸ	124
ሆነበ	125
ሆኖለሀ	158
ሆዲየ	165
ሆዳቸ	166
ሆዳይወ	167
ሆዴነ	168
ሆድሀ	169
ሆድሰ	170
ሆድሸ	171
ሆድወ	172
ለሌወ	178
ለልኸ	179
ለቀነ	220
ለቁመ	221
ለቃለ	222

Table 4-6: Conceptual lexicon

Then the semantic word lists that are found from preprocessing, by removing duplications and sorting it, changed to conceptual vocabulary. The conceptual vocabulary has a pointer to the posting file.

Pointers	Document	Locations	TFrequency
158	D10	[355, 1266, 1266]	3
165	D10	[1719, 1719]	2
166	D10	[314, 870]	2
158	D11	[20]	1
166	D11	[1155, 1155]	2
124	D13	[1676, 3068]	2
125	D13	[1139, 1139]	2
158	D13	[1482]	1
166	D13	[537, 537, 537]	3
.	.	[1344]	1

.	.	[250]	1
.	.	[312]	1
.	.	[2373]	1
		[1426]	1
		[1146, 1490, 1146]	3
		[124]	1
		[211, 695, 211]	3

Table 4-7: Posting File

4.4. LM Weighting and Customized VSM

Statistical weighting technique has its limitation because it is based on frequency model of Tf*Idf. But it is not only frequency that determine the Amharic term significance, inflection degree also determine the content bearingness of a term in the document. Because inflections in Amharic words hold information related with gender, tense and number.

This research incorporate inflection degree of a term in the document and in the overall corpus with the traditional Tf*Idf weighing scheme to determine the significance of a term in the document.

Tf*Idf determines how important a term is with respect to a document. The first idea in this weighting technique is, the more important a term is, the more often it appear which is called term frequency (tf).

$$tf_{t,d} = \sum_{x \varepsilon d} f_{t,}(x) \ where \ f_t(x) = \begin{cases} 1, & if \ x = t \\ 0, & otherwise \end{cases}$$

The weight of the term in the document calculated with the normalized weighting formula

$$W_{t,d} = \begin{cases} 1 + \log tf_{t,d}, & if \ \ tf_{t,d} > 0 \\ 0 & otherwise \end{cases}$$

Things to note in tf calculating formula, the first is order of terms appear in the document don't mater (ignored). The other is every terms in the document are equally important.

Terms occurring very often in the collection are not relevant for distinguishing among the documents. A relevance measure cannot only take term frequency into account. Inverse document frequency (Idf), rare terms across the document corpus are more important in explaining the document.

$$idf_t = log\frac{N}{df_t} \qquad with\ N = Collection\ size$$

During the experimentation it was able to visualize other factor that determines the importance of the term in the document. The information content of the term is always related with the degree of inflection in the document and/or in corpus.

For the special linguistic features of Amharic words as indicated in Example 4-1, even words that are considered as stop words can have meaning if there are many inflections related to the term in the document the term will have meaning related to the term.

To handle it another metric called inflection degree (Id) is developed. Inflection degree measures how many times the word form changed due to inflection in the document or in the overall corpus. If a word has a capability of inflecting in many different forms in the document, then there is more information related to that word [**17**].

$$Id_t = \frac{Inf_{t,d} * Inf_{t,C}}{tf_{t,c}}$$

$where$:

$-Inf_{t,d}: the\ inflection\ of\ the\ term\ in\ the\ documents$

$-Inf_{t,C}: the\ inflection\ of\ the\ term\ in\ the\ whole\ corpus$

$-\ tf_{t,c}$: $is\ the\ collection\ term\ frequency\ in\ the\ corpus$

The formula has two components, the first is Universal Significance, which determines the importance of the term with related with the total corpus.

$$U_{sig} = \frac{Inf_{t,C}}{tf_{t,c}}$$

The other part is local significance, which determines how the term is importance in the local document. Many non-content bearing words (like adverbs and prepositions) has very low of universal significance.

$$L_{sig} = \frac{Inf_{t,d}}{tf_{t,c}}$$

The normalized Id will be like,

$$Id_t = \log(\frac{Inf_{t,d} * Inf_{t,C}}{tf_{t,c}})$$

Id_t focuses which term is more explanatory of the document, the linguistically analyzer is supposed to identify the meaning of the inflected words. The resulting weight of a term is computed using tf, idf and Ir:

w(t, d) = tft,d × idft ×Id

- *high when t occurs many times in a small set of documents or appear in many documents with inflections*
- *low when t occurs fewer times in a document, or when it occurs in many documents without inflection*
- *Very low when t occurs in almost every document with the same form*

Score of a document with respect to a query can put as summation of weight of terms in the document which are found in the query.

$$w(q,d) = \sum_{t \epsilon q} w(t,d)$$

4.4.1. Customized Vector Space Model

In vector space model, documents and queries are arranged in n-dimensional vector space. The dimension of the vector space model is decided by the number of index terms. Even if the linguistic approach retain all the inflections of a term. Only conceptual vocabulary terms are used to construct Vector

space model. The other related words are referred during the time of searching.

Each term t of the dictionary is considered as a dimension. A document d can be represented by the weight of each dictionary term:

V~(d) = (w(t1, d), w(t2, d), . . . , w(tn, d)), for all documents in the corpus,

[D1[w_{t1}*,* w_{t2}*,* w_{t3}*,* w_{t4}*,* w_{t5}*,.... *w_{tn}*],*
D2[w_{t1}*,* w_{t2}*,* w_{t3}*,* w_{t4}*,* w_{t5}*,.... *w_{tn}*],*
D3[w_{t1}*,* w_{t2}*,* w_{t3}*,* w_{t4}*,* w_{t5}*,.... *w_{tn}*],*
D4[w_{t1}*,* w_{t2}*,* w_{t3}*,* w_{t4}*,* w_{t5}*,.... *w_{tn}*],*
.....
Dm[w_{t1}, w_{t2}, w_{t3}, w_{t4}, w_{t5},.... w_{tn}]]

Where w_{ti}*; represents the weight of the term at indicated index in the document*. The arrangements of terms for each document follow the order of conceptual lexicon. The python implementation will look like the posting file except it uses weights instead of occurrences.

4.5. Matching and Ranking

Cosine similarity is a measure of similarity between two vectors of an inner product space that measures the cosine of the angle between them. The cosine of 0° is 1, and it is less than 1 for any other angle. Two vectors with the same orientation have a Cosine similarity of 1, two vectors at 90° have a similarity of 0, and two vectors diametrically opposed have a similarity of -1, independent of their magnitude. Cosine similarity is particularly used in positive space, where the outcome is neatly bounded in [0,1].

Cosine similarity measure of how similar two documents are likely to be in terms of their subject matter.

The cosine similarity of any pair vectors is found by taking their dot product and dividing that by the product of their norms. That yields the cosine of the angle between the vectors.

$$cos\theta = \frac{d.q}{\|d\|\|q\|}$$

4.6. Evaluation of the system

The performance evaluation performed on both the preprocessor analyzer and the retrieval system. Preprocessor analyzer supposed to categorize a list of words given to it to the respective group of words according to syntactical analysis.

The effectiveness of the linguistic analyzer is measured how many of the total words in the corpus are grouped in the correct word group. This is done by recognizing the number of correctly predicted words over the number of total words in the corpus.

$$LA_{ef} = \frac{\{no\ of\ words\ correctly\ grouped\}}{\{total\ number\ of\ words\ in\ the\ corpus\}}$$

The retrieval model on the other hand there are IR performance evaluation techniques, such as, precision and recall, F-measure, E-measure, MAP (Mean average precision), R-measure. In this study, in this paper F-measure is used for the harmonic mean of Precision and recall.

Precision and recall are the two most frequent and basic statistical measures. Recall is the percentage of relevant documents retrieved from the database in response to users query, whereas precision is percentage of retrieved documents that are relevant to the query **[14]**.

Precision is the percentage of relevant from the retrieved list of documents

$$Precision = \frac{|\{relevant\ Documents\} \cap \{retrieved\ documents\}|}{|\{retrieved\ documents\}|}$$

Recall is the percentage of relevant documents that are effectively retrieved.

$$Recall = \frac{|\{relevant\ Documents\} \cap \{retrieved\ documents\}|}{|\{relevant\ documents\}|}$$

A measure that combines precision and recall is the harmonic mean of precision and recall, the traditional F-measure or balanced F-score:

$$F = 2 * \frac{Precision.Recall}{Precision + Recall}$$

This measure also called *F1* because precision and Recall are evenly weighted. It is the special case of F_β for non-negative real value β

$$F_\beta = (1 + \beta^2).\frac{Precision.Recall}{\beta^2.Precision + Recall}$$

CHAPTER FIVE EXPERIMENTATION

Linguistic preprocessor and retrieval model based on the linguistic preprocessor designed. Since there is no developed model for linguistic approach of IR system design, it needs to develop one and test its feasibility by testing multiple approaches to get the desired result for linguistic based retrieval system.

In this research an attempt made to develop the whole LM-IR system by designing its components. The following design components are chosen as the best choice for developing LM-IR in Amharic in this study.

To achieve the full LM-IR in Amharic, design

- A model that is appropriate for LM-IR in Amharic
- New approach of word interrelation using regular expression
- Linguistic analyzer (LA) using corpus based morph syntactic analysis. The output of this process is conceptual lexicon and extended word list related with the lexicon.
- Customized linguistically motivated indexes file (LMI) that is constructed using the linguistic method. This index file has extra linguistic information related to the inflection behavior of words in the language.
- A modified Matching and weighting technique for retrieval based on linguistic

This research used a total of **1180** text files that are Unicode encoded having Amharic on Unicode character encoding. These files are collected from the Amharic Holy bible of the year 1956 version by Birhana Selam. The naming sample for files is found in appendix III.

This corpus is used for both designing the Linguistic Analyzer and the retrieval system. Amharic bible can be considered as representative for Amharic word

variation, during the experimentation more than 74,000 unique words are found. Which means more than any dictionary of the language can provide, average dictionary of the language can have only 20,000 words.

The reason for specific domain corpus selection is to get high performance for LA using MSA. The analyzer can be more effective multiple word variation on single domain (deep) than collecting words from different domain at initial state.

According to J. Sparck [**31**] a successful and deep word analysis programs are those working with very limited domains. The narrow the domain of the corpus, the more effective the output of the analyzer would be.

Considering that, and taking Amharic Holy Bible as a primary source of the corpus will have the following advantage.

- ✓ Consistent Literature: the language used in the Holly Bible is formal and linguistically error free.
- ✓ Enough word variations: the bible can be representative of Amharic words; it can provide enough word variations of the single stem.

5.1. Test corpus preparation

The same corpus that used for building the LA also for experimentation LM-IR index files for construction. The files are 1180 which are taken from the 66 books of the holy bible. The version is the 1956 by Ethiopian Bible society.

Each chapter is stored in the form of text file by using the naming convention "[BookName][chapter#].txt". Then the chapters under every book are stored in a directory with the name of the book. Finally all the book directories are categorized into Old Testament and New Testament categories under the root directory called Holy Bible.

The encoding techniques used to store those chapter names is "UTF-8". UTF-8 is the universal and convenient encoding format for Unicode characters.

Unicode and Ethiopic/Amharic alphabets are discussed in detail on section 3.1.2. The analyzer program is capable of identifying Unicode encoded documents prior to preprocessing the documents inside the

- ✓ Compute all possible combination of words in the linguistically analyzer
- ✓ Capable of detecting prefix inflections and other derivations of the word

After tokenization of all the files in the corpus 415,451 words are found. The file size of the collected tokens written using Unicode list structure is 15MB.

Error! Reference source not found. Finally 74,045 unique of words are extracted from the total 415,451 words. This collection can serve as all possible words and their inflection found in the corpus.

The linguistic Analyzer was built on these words by using morph syntactic analysis (MSA). MSA determine the interrelationship of terms between each other using their morph. The detail of regular expression used for MSA described in section 4.2.

Based on regular expression processing on 74,045 words, 16,062 related groups are produced. These groups are supposed to be equivalent to performing statistical preprocessing on the corpus of files and getting representative vocabulary files. Each group is represented using conceptual vocabulary, that syntactic representation of all the words in the group

Linguistic approaches are known for their processing intensiveness, during experimentation, the first model took more than 365hrs, 42min and 7sec(which is morethan15 days) to finish the regular expression analysis on 64bit windows 7, Core i3 processor and 16GB ram. This is because each term needs 74,045x74,045x240 of computation.

For the effort of making it more computational efficient, it is optimized and reduced to 74,045x1,000x125 which took only 52hrs 11min and 4sec on the same machine.

5.2. Performance Evaluation LA

The performance of the LA for preprocessing is measure by the percentage of correctly classified word groups in the test set from the corpus of words. The comparison is also made with statistical preprocessing approach using the suffix shown in Table. This is because the prefix lists are very few and the same on both the LA and statistical approach.

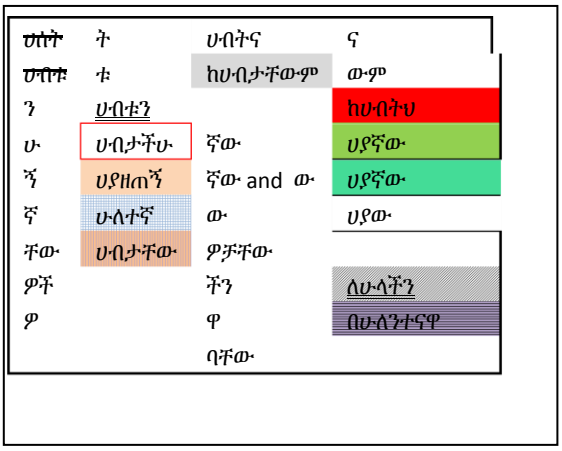
ኝ	ችን	ዎች	ዊት
ች	ቸው	ባቸው	ት
ና	ዊት	ቹን	ዬ
ዎች	ኛ	ዋ	ሁ
ዎ	ዎቻቸው	ላችሁ	በት
ባት	ቱ	ሽ	ይቱ
ዎችም	ዊያን	ሉ	ውያን
ነት	ችው	ዎ	ለት
ዎቻቸውም	ዎቹ	ያዊ	ዊ
ህ	ላት	ባቸው	የው
ውም	ናቸው	ን	ዊቷ
ላቸው	ባችሁ	ኞች	

Table 5-1: Suffix list that the statistical approach uses

After the performance test using the experimentation model on the first 5500 of 74,045 words, the linguistic preprocessing by far outperform the statistical approach, in identifying the words proper category by performing MSA described in section 4.2. Appendix XIII shows more than 3500 list of words and their parallel competition.

How the performance of the LA and statistical preprocessing measured.

S.N	word	MSA CV	Statistical root word	
1	ሀሩፋዊው	ሀሩፋወ	ሀሩፋዊው	
2	ሀሮኤ	ሀሮአ	ሀሮኤ	
3	ሀሰት	ሀሰተ	~~ሀሰት~~	wrong w
4	ሀበቱ	ሀበተ	~~ሀበቱ~~	wrong w
5	ሀበቱም	ሀበተ	ሀበቱም	1(-)
6	ሀበቱን	ሀበተ	ሀበቲን	2(ን)
7	ሀበቱንም	ሀበተ	ሀበቱንም	3(-)
8	ሀበታም	ሀበተ	ሀበታም	4(-)
9	ሀበታቸው	ሀበተ	ሀበታቸው	5(ቸው)
10	ሀበታቸውን	ሀበተ	ሀበታቸውን	5 or 6(ን)
11	ሀበታቸውንም	ሀበተ	ሀበታቸውንም	7(-)
12	ሀበታችሁ	ሀበተ	ሀበታችሁ	5(ችሁ),8(ሁ)
13	ሀበቴ	ሀበተ	ሀበቴ	9(-)
14	ሀበት	ሀበተ	~~ሀበት~~	wrong w
15	ሀበትና	ሀበተ	ሀበትና	10(ና)
16	ሀበትን	ሀበተ	ሀበትን	10(ን)



17	ሀብትንም	ሀብተ	ሀብትንም	11(ንም)
18	ሀብትንና	ሀብተ	ሀብትንና	12(ና)
19	በሀብታቸው	ሀብተ	በሀብታቸው	5(ቸው)
20	ከሀብታቸውም	ሀብተ	ከሀብታቸውም	13(ውም)
21	ከሀብትህ	ሀብተ	ከሀብትህ	10(ህ)
22	ከሀብትህም	ሀብተ	ከሀብትህም	14(-)
23	የሀብቱን	ሀብተ	የሀብቱን	2(ን)
24	የሀብት	ሀብተ	~~የሀብት~~	wrong w
25	ሀኪተኖችም	ሀኪተኖች	ሀኪተኖችም	1(-)
26	የሀኪትንም	ሀኪትነ	የሀኪትንም	2(-)
27	ሀያ	ሀየ	ሀያ	1(-1)
28	ሀያም	ሀየ	ሀያም	2(-)
29	ሀያኛው	ሀየ	ሀያኛው	1(ኛው),3(ው)
30	ሀያው	ሀየ	ሀያው	1(ው)
31	ሀያውን	ሀየ	ሀያውን	4(ን)
32	ሀያዘጠኝ	ሀየ	ሀያዘጠኝ	5(ኝ)

Table 5-2: LA vs statstical preprocessor results

From the table let's take the term ሀብት "habt" which means rich and let's see the difference between statistical preprocessing stemming and linguistic MSA to determine the word groups.

The statistical approach uses suffixes listed **Error! Reference source not found.** to remove from the word. Using this technique delivers a very disappointing result of statistical preprocessing. From total 20 words that are related the results are in

words	Types by statistical words		Words at position
3	wrong terms		4,14,24
12	different roots		5-11,13,15-18,20,22
3	Related vocabulary words	With 6	23,
		With 9	10, 19
		With 21	16,20

Table 5-3: effectiveness of the statistical preprocessing example

The effectiveness of the approach, is calculated as

$$Accuracy = \frac{total\ words - incorrectly\ classified}{\#of\ groups * totalwords} * 100\%$$

$$Statstical\ Accuracy = \frac{(20 - (3 + 12))}{3 * 20}$$

$$Statstical\ Accuracy = 8\%$$

This result will show further performance degradation for words that have more inflection or words which have more character in the word.

On the other hand the LA predicted the conceptual vocabulary to be ሀብተ for all the words related to the group. That means the conceptual vocabulary can be representative all related words and could group into one set as representative of the other lists. Measuring the accuracy getting the root word from the inflected words the LA outperform the statistical approach.

$$LA\ Accuracy = \frac{(20 - 0)}{1 * 20} * 100\%$$

$$LA\ Accuracy = 100\%$$

Interesting relation with the number of characters in a word and the accuracy of linguistic analyzer for preprocessing is visualized.

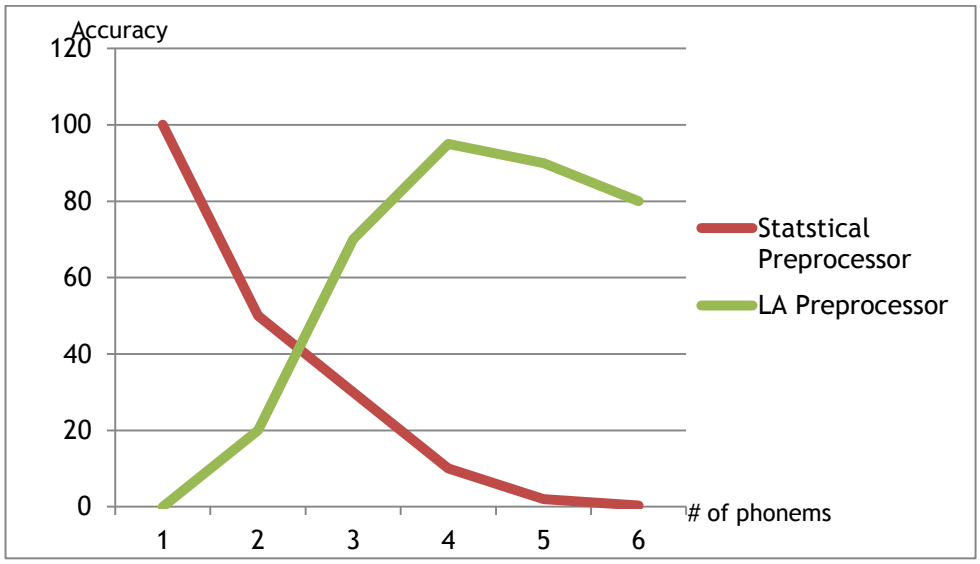


Figure 5-1: effectiveness of statistical Vs. LA preprocessing with the no. of phonemes

The linguistic preprocessor prediction capability reaches pick when a root words contain 3 to 4 phonemes. But statistical preprocessor approach fails to predict the correct root word while the number of characters in the word increases.

One other advantage of LA preprocessor is its use for multi-level indexing. Words preprocessed with LA show incremental inflection, this helps to find the next related meaning immediately the one word group.

From Appendix V we can see that the consecutively related words are found next to each other. This file architecture helps to design index file structure that improves both recalls and precision at the same.

It is also possible to use this feature to determine word parts in the word by performing in-depth analysis of the conceptual vocabulary itself. Since Amharic is morphological, and those morphs contain information related to important parts of word of the language.

This was the very challenge of Amharic retrieval system. In this research the conceptual vocabulary has this capability to determine these properties of the word simply.

Conceptual voc.	behavior	Conceptual voc.	behavior
ሄደ	The main root	ሄዱተ	3rd pers, plural,
ሄጃለሀ	1st person, passtense	ሄዱበ	3rd pers, plural, obj
ሄጀ	1st person	ሄዱመ	
ሄዶአለ	3rd pers, masculine, past	ሄደወ	
ሄድነ	1st pers, plural	ሄደእንዲሀ	
ሄድበ	3rd pers, sing, obj	ሄደቸ	
ሄድሽ	2nd pers, past, feminine	ሄደበ	
ሄድሀ	2nd pers, past, masculine	ሄደመ	
ሄዳችኋለ	2nd pers, plural, past	ሄደሀ	
ሄዳችሀ	2nd pers, plural,	ሄደለተ	

Table 5-4: conceptual vocabularies for multiple levels

Using LM-Index file it's possible to retrieve respective to linguistic behavior of the language. If more precise information related to the gender, tense and other word parts needed, the query term preprocessed to match the longest conceptual vocabularies string, else if recall oriented is needed, the query terms should be changed to the smallest possible term in the conceptual vocabulary.

This approach is handy especially for words that are related yet having different inherent meaning. For example words ቅዱስ(Holy) and ቅድስና(Sanctifying) are related terms; but this words don't cross each other, one cannot inflect into the other territory. For this case the three term conceptual vocabulary can differentiate

The following terms are related with ቅዱስ

ለቅዱሱም	ቅዱስ	በቅዱስ
ለቅዱስ	ቅዱስህን	በቅዱስም
ለቅዱስህ	ቅዱስህንም	በቅዱስነቱ
ለቅዱስተ	ቅዱስም	በቅዱስነቴ
ቅዱስ	ቅዱስነት	ከቅዱስ
ቅዱሱ	ቅዱስነትና	ከቅዱሱ
ቅዱሱም	ቅዱስና	ከቅዱስ
ቅዱሱን	ቅዱስን	ከቅዱስነትህና
ቅዱሱንም	ቅዱስንም	የቅዱሱን
ቅዱሳት	በቅዱሱ	የቅዱሱንና
ቅዱሳችሁ	በቅዱሱም	የቅዱስ

Table 5-5፡ terms related to the term ቅዱስ

The following terms are related with ቅድስና

ለቅድስና	ቅድስናና	በቅድስናውስ
ለቅድስናም	ቅድስናን	ከቅድስናው
ለቅድስናውም	ቅድስናዬ	የቅድስና
ቅድስና	በቅድስና	የቅድስናህን
ቅድስናህ	በቅድስናም	የቅድስናህንም
ቅድስናም	በቅድስናና	የቅድስናና
	በቅድስናው	የቅድስናውን

Table 5-6፡ terms related to the term ቅድስና

The statistical approach cannot resolve issue related to the concept and meaning during preprocessing. The advantage of linguistic preprocessing over statistical approach is overwhelming.

5.3. Performance Evaluation of LM-IR

The performance of the retrieval part of LM-IR affected by not only the indexing structure and vocabulary file, but also the type of weighting and matching technique applied.

To test the performance of the designed system, different type of queries were used. Those queries vary in inflection degree; some queries hold words that

cannot be inflected for instance name of a place or person and highly inflected words.

The multi-level Linguistic index enables to improve recall without affecting the precession. This index structure can be approached to design a system that is both recall and precession at the same time. This experimentation uses different queries to test these performances.

The searching result shows us almost all relevant documents can be retrieved without affecting the precision of the system; this might be great use for general purpose IR system design. To make a system more precise the deeper level of the index structure can be used. This precision oriented on the other hand will retrieve specific information need of the user for the case gender, number and other grammatical inflections are highly considered.

S.N	Query	RelV. Docs	Retrvd. Docs
1	በሰርዴስም ወዳለው	293, 579	293, 579
2	የኬጢያውያን ምድር	*781, 1095, 61, 251, 598, 70, 220, 576, 758, 165, 417, 284, 917, 51, 772, 399, 280, 1007, 420*	*781, 1095, 61, 251, 598, 70, 220, 576, 758, 165, 138, 282, 417, 284, 917, 51*
3	እንደሚንሸዋሸዋ ጸናጽል ሆኜአለሁ	1167, 608, 627, 222, 293	1167, 608, 627, 222, 293, 147
4	ባትነቃ እንደ ሌባ እመጣብሃለሁ	584, 790, 972, 1124, 293, 25, 151, 604, 708, 1016, 807, 930, 267, 472, 203, 1177, 558, 321, 913, 947	25, 86, 87, 151, 172, 267, 293, 461, 480, 558, 584, 604, 693, 708, 790, 807, 885, 892, 930, 945
5	ከእግዚአብሔር ጋር ታግለህ	320, 1167	320, 1167, 53, 558
6	ሰውም ከእግዚአብሔር አፍ በሚወጣ ነገር ሁሉ በሕይወት እንዲኖር እንጂ	510, 467, 414, 140, 960, 811, 238, 167, 815, 599, 1102, 1150, 478, 1024, 282, 552, 959, 803, 703, 484, 120	510, 467, 414, 960, 140, 167, 811, 238, 1102, 1150, 478, 12, 484, 325, 817, 1152, 990, 620, 1024, 282
7	የሐዋርያት ሥራ	249, 71, 193, 714, 1099, 667, 655, 135, 360, 404, 98, 85, 311, 751, 628, 1006, 42, 456, 40, 96, 1064, 400, 318, 242, 50, 423, 779, 351, 181, 635, 709, 681, 394, 315, 685, 91, 246, 277, 349, 436, 449, 943, 1098, 131, 875, 155, 728, 429, 147, 1114, 407, 430, 15, 416, 679, 134, 172, 270, 966, 998, 228, 1079, 1083, 1102, 107, 34, 276, 912, 12,123, 171, 410, 427, 478, 696	249, 71, 193, 714, 271, 1119, 1070, 1099, 758, 667, 655, 135, 360, 404, 98, 85, 311, 751, 628, 1006, 42, 456, 40, 96, 1064, 400, 318, 242, 50, 423, 779, 351, 181, 635, 709, 681, 394, 315, 685, 91, 246, 277, 349, 436, 449, 943, 1098, 131, 875, 155, 728, 429, 147, 1114, 407, 430, 15, 416, 679, 134, 172, 270, 966, 998, 228, 1079, 1083, 1102, 107, 34, 276, 912, 12,

8	በደሙ የተደረገ ቤዛነታችን	722, 416, 228, 78, 779, 1079, 992, 481, 276, 592, 265, 664, 845, 994, 1035, 571, 117	741, 722, 416, 228, 78, 1066, 794, 779, 1079, 992, 481, 276, 592, 265, 664, 845, 994, 1035, 571, 117
9	ታላቁም ካህን የኢዮሴዴቅ	*43, 51, 129, 254, 437, 820, 927, 1172*	*43, 51, 129, 254, 437, 820, 927, 1172*
10	በአምላካችንና በመድኃኒታችን በኢየሱስ ክርስቶስ ጽድቅ	*667, 641, 623, 1104, 427, 493, 1062, 895, 438, 667*	*1135, 667, 641, 623, 1104, 1137, 427, 650, 493, 1062, 780, 895*

Table 5-7: Test query, relevant docs and ranked output

The LM-IR performance measure is done using precision, recall and F-measure on experimentation query.

The table below shows below shows the result by using multi-level indexing constructed on the corpus.

S.N	Query	Rel	Rtrvd	Rel Rtrvd	P	R	F
1	በሰርዴስም ወዳለው	2	2	2	1.0	1.0	1.0
2	የኬጢያውያን ምድር	19	20	18	0.9	0.95	0.92
3	እንደሚንሸዋሸዋ ጸናጽል	5	6	5	0.83	1.0	0.91
4	ባትነቃ እንደ ሌባ እመጣብሃለሁ	25	22	15	0.68	0.6	0.64
5	ከእግዚአብሔር ጋር ታግለህ	2	4	2	0.50	1.0	0.66
6	ሰውም ከእግዚአብሔር አፍ በሚወጣ ነገር ሁሉ በሕይወት እንዲኖር እንጂ	21	21	14	0.66	0.66	0.66
7	የሐዋርያት ሥራ	76	74	70	0.95	0.92	0.93
8	በደሙ የተደረገ ቤዛነታችን	20	19	17	0.95	0.85	0.90
9	ታላቁም ካህን የኢዮሴዴቅ	7	7	7	1.0	1.0	1.0
10	በአምላካችንና በመድኃኒታችን በኢየሱስ ክርስቶስ ጽድቅ	10	12	8	0.66	0.8	0.72
				Average	0.76	0.878	0.83

Table 5-8: The performance of the system on test queries

From the Table 5-8 we can see that query 1 and 9 have maximum precession. This is because the query phrase contains non inflecting words የኬጢያውያን and የኢዮሴዴቅ. These words are non-inflected and found in few documents therefore the retrieval system capable of retrieving all documents related to it.

The result that query 2 don't have complete precision is because it is attached with the terms ምድር that have high inflection rate even to the deeper level.

Generally, if a query phrase contains highly inflecting terms with in it the recall will increases, the case of query 3 and 5, on the other hand if the query phrase contain a unique word or highly inflected word the more precise results will obtained.

5.4. Findings and Challenges

The accuracy of linguistic analyzer outperforms the statistical approach in preprocessing document and query terms. Beside the accuracy of predicting word group, the LA avoids the semantic degradation of inflected words during preprocessing. Semantic degradation has been the great challenge on Amharic word preprocessing due application of suffix and prefix list for stemming.

The LA can predict the words with 3-4 phonemes root words at accuracy of 90% irrespective of how many different ways a word inflected. This is by far unreachable for statistical approach of preprocessing.

The indexing structure of multi-level indexing can provide an efficient retrieval approach that can increase recall without the coast of precision. This index file structure is the result of LA, which can provide multiple level of word analysis to keep the related information with inflection like gender, number, tense or any word part. To see multiple level of word analysis, take a look at Table 5-4 and Table 5-5. Amharic retrieval that is built on the LMI can deliver better result than the statistical approach. The retrieval effectiveness is the result of proper categorization of related (inflected) terms to construct multi-level indexing. The result of better performance is because of the nature of Amharic language and its writing system.

The one challenge related with LA is, the performance meltdown when the phoneme of a word reduced. This is due to the technique applied for grouping related words is a regular expression. The lower the number of phonemes is the word the higher the number of matches from the word list, also the higher the number of unrelated words grouped together.

CHAPTER SIX CONCLUSION AND RECOMMENDATIONS

Information Retrieval is a very essential tool for a society by providing access to relevant information in a fast and convenient way. Although the use of IR left behind societies like Ethiopia by widening the digital divide with technologically established ones. The only option to narrow this digital divide is developing effective and efficient IR systems that benefit the local language users of the society.

6.1. Conclusion

The goal of this study is exploring a new linguistic approach of effective and efficient retrieval on Amharic documents. This study completed three major tasks; identifying key Amharic linguistic features, designing LA and developing customize weighting technique.

The identified features of Amharic language that should be considered before designing any IR system are syllabic alphabetic encoding and morphological nature of word formation. Any model that used on Amharic IR system must focus on these two major characteristics of the language. In general Amharic IR system should be linguistics first.

The morphological nature of Amharic language makes it difficult to apply stemming algorisms. On this study we managed it by designing a new Linguistic Analyzer (LA) in place of preprocessor steps. LA is word preprocessor that uses Morph syntactic Analysis (MSA) to detect linguistic variation and group related (inflected) words to their conceptual root term. This research uses conceptual root term is used to represent the family of inflected words.

The other thing this paper contribute is developing LA for Amharic word preprocessing which enable multilevel indexing that resolves the challenge of meaning loose during grouping of related words. Using Multi-level indexing it's

possible to keep information related with gender, number, and tense. This helped resolve linguistic ambiguity and linguistic variation challenges.

The effectiveness of LA compared with statistical preprocessing techniques on Amharic words is remarkable. During experimentation done on 5000 words the LA delivers more than 80% of accuracy, whereas the statistical stemming cannot perform below 30% on the same corpus. The LA preprocessor results can even reach up to 90% when the numbers of characters in the root word are 3 to 5.

The experimentation result shows a 76% precession, 88% recall and their harmonic mean 83%. This is due to a customized weighting Tf*Idf technique applied. However it should be tested compared with the statistical IR approach that uses preprocessing like stemming and stop word removal to see the potential benefits of the Linguistic approaches of IR over the statistical approach.

6.2. Further Research Directions

This research will create a new door for design and implementation of Amharic IR from linguistic perspective. Further researches needs to be done on identifying the linguistic factors of Amharic that determine its word formation.

- Detail study on Amharic to search for other linguistic factor of Amharic and append it to the current Syntactic Analysis (MSA) to enhance LA can further improves the effectiveness of the retrieval system
- Further research at sentence level Morph MSA will help in finding a new approach of sematic interrelation between words of the sentence. It will enhance IR in Amharic
- Integrate linguistic analysis technique with other statistical retrieval models and technique will help to find further improved Amharic LM-IR.

- Designing more intelligent Regular Expression to enable detect the right groups of words that are formed with similar phonemes but different characters for searching will help to enhance the performance of the system further
- Using Python 3.X series to design linguistically motivated IR system will create a better opportunity to handle Unicode operations and increase efficiency of the retrieval system
- Perform re-ranking on the fly to make the system more effective while retrieving.

References

1. Sanderson, M., Croft, W.: The History of Information Retrieval Research., Melbourne (2012)

2. Mooers, C.: Zatocoding applied to mechanical organization of knowledge. American Documentation, 2, , 20-32 (1951)

3. Yang, C. C., Chen, H., Honga, K.: Visualization of large category map for Internet browsing., 89–102. (2003)

4. Taube, C.: Unit terms in coordinate indexing. American Documentation, vol. 3, no. 4, pp. 213-218 (1952)

5. Cleverdon, C.: The Evaluation of Systems Used in Information Retrieval (1958: Washington). Proceedings of the International Conference on Scientific Information -- Two Volumes, 687-698 (1959)

6. Rijsbergen, C.: INFORMATION RETRIEVAL. University of Glasgow, Glasgow (1979)

7. Mooney: Information Retrieval(course slides). In: University of Austin. (Accessed 2005) Available at: http://www.cs.utexas.edu/user/money/ir-course

8. Ingwersen, P.: INFORMATION RETRIEVAL INTERACTION. Taylor Graham Publishing (2002)

9. Vallez, M., Pedraza-Jimenez, R.: Natural Language Processing in Textual Information Retrieval and Related Topics. In: ww.Hipertext.net. (Accessed June 5, 2012) Available at: http://www.upf.edu/hipertextnet/en/numero-5/pln.html

10. Sanderson: Information Retrieval. In : Retrieving with good sense. (2000) 49-69

11. Baeza-Yate: Challenges in the interaction of Information retrieval and Natural Language processing. In : International Conference on Computational Linguistics, Seol, p.Intelligent Text Processing (2004)

12. Bizuneh, B.: THE APPLICATION OF WEBSOM FOR AMHARIC TEXT RETRIEVAL., Addis Ababa (2003)

13. Alemayehu: Application of Query Expansion for Amharic information retrieval System., Addis Ababa (2002)

14. Amanuel: Probablistic Information Retrieval., Addis Ababa (2012)

15. Bethlehem, M.: The Application of N-gram Indexing in Amharic Text Retrieval., Addis Ababa (2002)

16. Saba, A.: The Application of Information Retrieval Technique to Amharic documents in the web., Addis Ababa (2001)

17. Getahun, A.: ዘመናዊ የአማርኛ ሰዋሰው በቀላል አቀራረብ (Moder Amharic Grammer in Simplified way). Alpha Printers, Addis Ababa (2010)

18. Gezehagn, G.: Afaan Oromo Text Retrieval System., Addis Ababa (2012)

19. Baeza-Yate, R., Ribeiro-Neto, B.: Modern Information Retrieval. ACM Press, New York (1999)

20. Plone. In: http://quintagroup.com/. (Accessed 2012) Available at: http://quintagroup.com/cms/python/ide

21. Bruck, A.: Semantic Based Query Expansion Technique for Amharic IR, MSc Thesis, School of Information Science,. (2011)

22. Harter, S.: Online information retrieval.. Academic Press, Inc., Orlando,Florida (1986)

23. INGWERSEN, P.: INFORMATION RETRIEVAL INTERACTION. Taylor Graham Publishing (2002)

24. Anwar, A.: web Information Retrieval and Search Engine Technique. Al-Satil journal, 55-92 (2008)

25. Tomek Strzalkowsky[Editer]: Kluwer Accadamic Publisher: Natural Language Information Retrieval. Kluwer Accadamic Publisher, AH Dordrecht, Netherlands (1999)

26. Timo, L.: Automatic indexing: an approach using an index term corpus and combining linguistic and statistical methods., Helsinki (2000)

27. Hiemstra, D.: Information Retrieval Models., New york (2009)

28. American National Standards Institutes.: Basic Criteria for Indexes. ANSI Z39.4, New York (1968)

29. Fasika, T.: Phrasal translation fro Amharic English Cross Lnaguage IR. Addis Ababa University, Addis Ababa (2010)

30. Meshesha, M.: Advanced Information Retreival lecture note. Addis Ababa University, Addis Ababa (2012)

31. Jones, S., J.I, T.: LINGUISTICALLY MOTIVATED DESCRIPTIVE TERM SELECTION. Computer Laboratory, University of Cambridge, Corn Exchange Street, Cambridge CB2 3QG, U.K. (1999) 287–290

32. Karen, S.: What is the role of NLP in text retrieval? Natural Language Information Retrieval 1(24) (1997)

33. Strzalkowski, T.: Natural Language Information Retrieval. Computational Linguistics Volume 26(Number 3), 460 - 462 (1999)

34. Foskett, A. J.: The Subject Approach to Information. 2. ed. Bingley, London (1971)

35. Knuth: Retrieval On secondary Keys. Moffat & Ramamohanarao, Zobel (1998)

36. http://www.mec.ac: Indexed sequencial files. Available at: http://www.mec.ac.in/resources/notes/notes/ds/saf.htm

37. Croft, W.: TSearch engines: Information retrieval in practice. Addison-Wesley, 283 (2010)

38. www.csee.umbc.edu: Searching with inverted file., www.csee.umbc.edu (2012)

39. Christodoulakis, C.: Signature files. An access method for documents and its analytical performance evaluation. ACM Transactions on Information Systems (TOIS) 2(4), 267-288 (1984)

40. Carterette, B., Can, F.: Comparing Inverted Files and Signature Files for Searching a Large Lexicon., Miami (2003)

41. Abouelhoda, M., Kurtz, S., Ohlebusch, E.: Replacing suffix trees with enhanced suffix arrays. Journal of Discrete Algorithms 2, 53 (2004)

42. Aluru, S.: Suffix Trees and Suffix Arrays., Iowa State

43. Turtle, H., croft, W.: A comparision of Text Retrieval Models. In : West Publishing company; University of Masachuset, Masachuset, USA (1991)

44. Salton, G., Buckley, C.: Term Weighting approach in Automatic Text

Retrieval., New York (1987)

45. Cambridge University: Scoring, Term wighting & the vector space model. Cambridge University Press (2009) 109-133

46. Fagan, J. L.: Experments in Automatic Phrase Indexing for Document Retrieval: A comparision of Syntactic and Non Syntactic Methods. A PhD thesis., Cornell University

47. Smeanton, A. F., C.J., V.: Experments on Incorporating Syntactic Processing of user queries into document Retrieval strategy. In : the 11th International ACM-SIGIR conference on Research and Development in Information Retrieval , Grenoble, France (1988)

48. Brants, T.: Natural Language Processing in Information Retrieval., California

49. Voutilainen, K., A., H., , A.: Constraint Grammar: A language Indipendent System for Parsing Unrestricted text:., Berlin & NewYork

50. Smeaton, A.: Using NLP or NLP Resources fo Information Retrieval Tasks. Natural Language Information Retrieval, 99-111 (1999)

51. Wendlandt E.B. & Driscoll, J. R.: Incorporating a semantic analysis into document retrieval strategy. In: A. Bookstein et.al. In : 14th AMC/SIGIR Conference, Chicago (1991)

52. Theodros, H.: Automatic Text Retrieval: An Experimenting using Latent Semantic Indexing (LSI) with Singular Value Decomposition (SVD)., Addis Ababa (2003)

53. Solomon, A.: Design and Implementation of Amharic Search Engine, fifth international conference on signal image technology and internet based systems., addis Ababa (2009)

54. Mamuye, B.: The application of Websome for Amharic text., addis Ababa (2003)

55. Solomon, T.: Design and Implementation of Amharic Search Engine. fifth international conference on signal image technology and internet based systems 3(1), 318-325 (2009)

56. Amanda, W., Rachel, W.: Amharic Language & Culture Manual., Texas (2011)

57. Letta, A.: Feature Extraction and Matching in Amharic Document Image Collection., Addis Ababa (2011)

58. M.L, B., Head, S., Cowley, R.: The Ethiopian Writing system, Language in Ethiopia. (1976)

59. Gippert, J.: Encoding before and after Unicode: The case of Ethiopic and Amharic., Hamburg (1997)

60. Jensen, H.: Sign, Symbol and Script: An Account of Man's Efforts to Write. Putnam, the University of California (1969)

61. McMillan, K.: unicode in python, completely demistified. (Accessed 2008) Available at: http://farmdev.com/talks/unicode/

62. Spolsky, J.: Joel on Software. (Accessed October 8, 2003) Available at: http://www.joelonsoftware.com/articles/Unicode.html

63. Klavans, J., Jacquemin, C.: A Natural Lnaguage Approach to Multi-Word Term Conflation. Bell Laboratories, Lucent Technology, Murray Hill,USA (1997)

64. Zhai, C.: Axiomatic analysis and optimization of information retrieval models. In Proceedings of the Third international conference on Advances

in information retrieval theory. Springer-Verlag., 1 (2011, September)

65. Winograd: Understanding of matural language. Edinburgh University Press, Edinburgh (1972)

66. Manning, C.: Introduction to information retrieval (Vol. 1). Cambridge University Press, Cambridge (2008)

67. Korfhage, R.: Information storage and retrieval. (2008)

68. Hjørland, B.: Core Concepts in Library and Information Science. In: http://www.iva.dk. (Accessed October 15, 2006) Available at: http://www.iva.dk/bh/Core%20Concepts%20in%20LIS/articles%20a-z/information_retrieval.htm

69. Goker, A. .: Information retrieval: searching in the 21st century. Wiley (2009)

70. Fernandes, J.: Recent advances in IR-UWB transceivers: An overview. In Circuits and Systems (ISCAS). Proceedings of IEEE International Symposium on, 3284-3287 (2010)

71. Deerwester, S. .: Indexing by latent semantic analysis. Journal of the American society for information science, 41(6), 391-407 (1990)

72. Croft, W.: Search engines: Information retrieval in practice. Addison-Wesley, 283 (2010)

73. Büttcher, S. .: Information retrieval: Implementing and evaluating search engines. The MIT Press (2010)

74. Brier, S.: Cybersemiotics: Why Information is Not Enough! University of Toronto Press., Toronto (2008)

75. Cambrigde University Press: http://www.cambridge.org/. In: http://www.cambridge.org/. (Accessed 2009) Available at: http://www.cambridge.org/home/home/item5655304/?site_locale=en_US

76. Gross, B.: The Managing of Organizations: The Administrative Struggle. (1964)

77. Baye, Y.: የአማርኛ ሰዋሰው. ት.መ.ማ.ማ.ድ, Addis Ababa (1995)

78. RIJSBERGEN, C.: INFORMATION RETRIEVAL. University of Glasgow, Glasgow (1979)

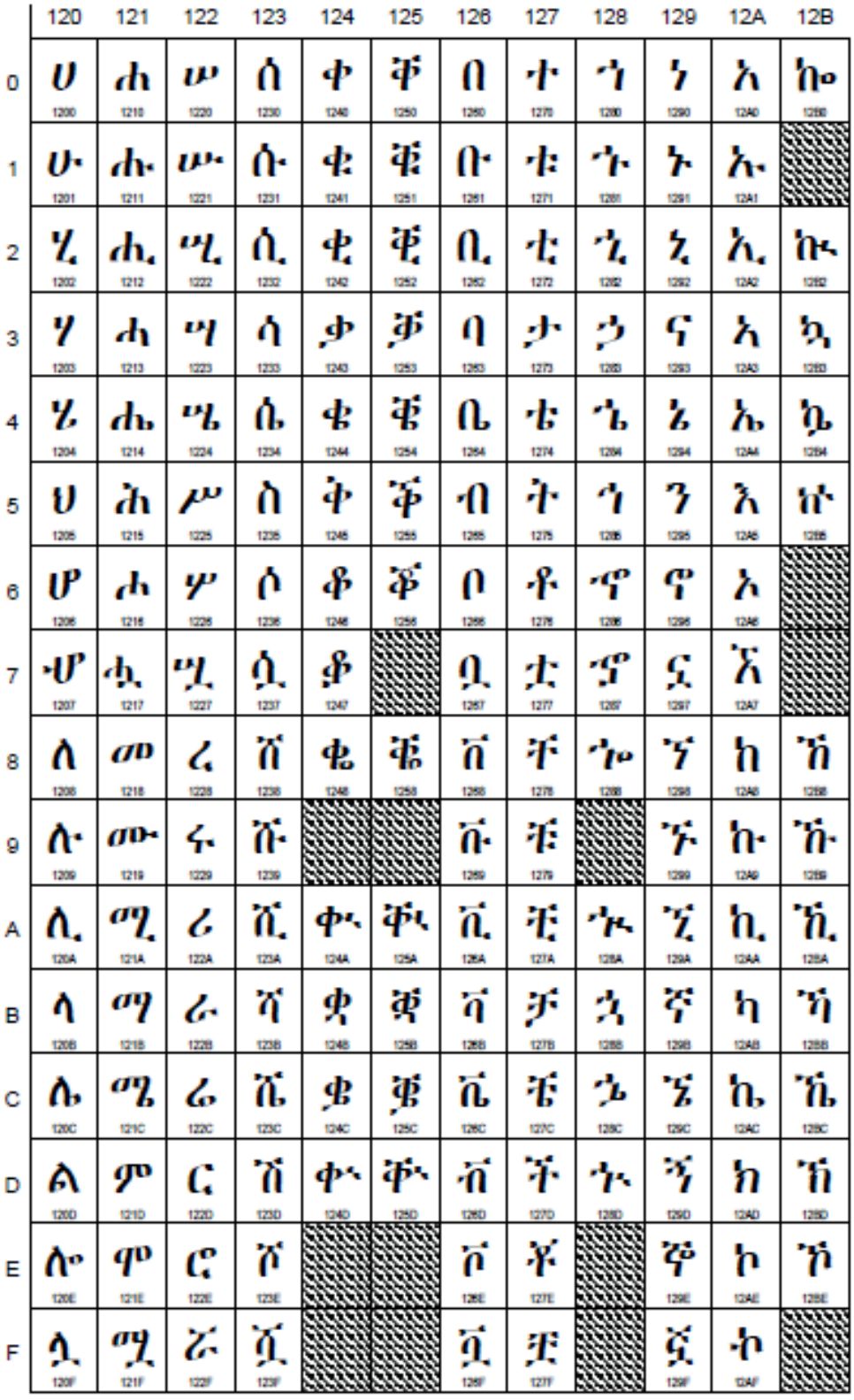
79. M. Taube, C.: Unit terms in coordinate indexing. American Documentation, vol. 3, no. 4, pp. 213-218 (Jan 1952)

80. FOSKETT, A. J.: The Subject Approach to Information. 2. ed. Bingley, London (1971)

81. Manning: Introduction to Inforamtion retrieval. Cambridge University Press, New York (2008)

82. McMilan, K.: Unicode In Python, Completely Demystified. (Accessed 2008) Available at: https://farmdev.com/talk/unicode/

Appendix I: Ethiopic Unicode standard from 1200 to 137F Source: www.unicode.org, 2012



Appendix II: Stop word list of Amharic Source: Probabilistic IR, by Ammanuel [14]

ነው	እኔ	እኛ	እነሱ	እሱ	እሷ	አንተ	እናንተ	እና	ወደ
ነይ	ወይ	ከ	ናቸው	ትናት	ጥቂት	በርካታ	ብቻ	ሁሉም	ሌላ
ሌሎች	ሁሉ	እያንዳንዱ	እያንዳንዳቸው	ስለ	እንዲሁም	እንጂ	ደግሞ	መካከልከ	ሰሞኑን
ከሰሞኑ	በሰሞኑ	የሰሞኑ	ትናንት	ትናንትና	ጋራ	የጋራ	ከጋራ	ተለያዩ	ተለያየ
ድረስ	እስከ	በጣም	ግን	ሲሆን	ሲል	ዉስጥ	ላይ	ናት	ነበሩ
ነበረች	ያ	ወይዘሮ	ወይዘሪት	ነገሮች	ከፊት	ከላይ	ታች	ከታች	በታች
የታች	በውስጥ	ከውስጥ	ጋር	ናቸው	ይህ	በላይ	ወደ	ወዘተ	እና
ወይም	እንደ	አቶ	ፊት	ወደፊት	ነገር	በፊት	በሆላ	በኩል	

Appendix III: File name naming formats samples

S.N	File Name	S.N	File name	S.N	File Name
1	1coronthians1.txt'	48	2Coronthians7.txt'	95	Ephesians1.txt'
2	1coronthians10.txt'	49	2Coronthians8.txt'	96	Ephesians2.txt'
3	1coronthians11.txt'	50	2Coronthians9.txt'	97	Ephesians3.txt'
4	1coronthians12.txt'	51	22John1.txt'	98	Ephesians4.txt'
5	1coronthians13.txt'	52	1peter1.txt'	99	Ephesians5.txt'
6	1coronthians14.txt'	53	1peter2.txt'	100	Ephesians6.txt'
7	1coronthians15.txt'	54	1peter3.txt'	101	Gelatians1.txt'
8	1coronthians16.txt'	55	2Tesolenian1.txt'	102	Gelatians2.txt'
9	1coronthians2.txt'	56	2Tesolenian2.txt'	103	Gelatians3.txt'
10	1coronthians3.txt'	57	2Tesolenian3.txt'	104	Gelatians4.txt'
11	1coronthians4.txt'	58	2Timoti1.txt'	105	Gelatians5.txt'
12	1coronthians5.txt'	59	2Timoti2.txt'	106	Gelatians6.txt'
13	1coronthians6.txt'	60	2Timoti3.txt'	107	Hebrew1.txt'
14	1coronthians7.txt'	61	2Timoti4.txt'	108	Hebrew10.txt'
15	1coronthians8.txt'	62	3John1.txt'	109	Hebrew11.txt'
16	1coronthians9.txt'	63	Acts1.txt'	110	Hebrew12.txt'
17	1John1.txt'	64	Acts10.txt'	111	Hebrew13.txt'
18	1John2.txt'	65	Acts11.txt'	112	Hebrew2.txt'
19	1John3.txt'	66	Acts12.txt'	113	Hebrew3.txt'
20	1John4.txt'	67	Acts13.txt'	114	Hebrew4.txt'
21	1John5.txt'	68	Acts14.txt'	115	Hebrew5.txt'
22	1Peter1.txt'	69	Acts15.txt'	116	Hebrew6.txt'
23	1Peter2.txt'	70	Acts16.txt'	117	Hebrew7.txt'
24	1Peter3.txt'	71	Acts17.txt'	118	Hebrew8.txt'
25	1Peter4.txt'	72	Acts18.txt'	119	Hebrew9.txt'
26	1Peter5.txt'	73	Acts19.txt'	120	James.txt'
27	1Tesolenian1.txt'	74	Acts2.txt'	121	James2.txt'
28	1Tesolenian2.txt'	75	Acts20.txt'	122	James3.txt'
29	1Tesolenian3.txt'	76	Acts21.txt'	123	James4.txt'
30	1Tesolenian4.txt'	77	Acts22.txt'	124	James5.txt'
31	1Tesolenian5.txt'	78	Acts23.txt'	125	John1.txt'
32	1Timoty1.txt'	79	Acts24.txt'	126	John10.txt'
33	1Timoty2.txt'	80	Acts25.txt'	127	John11.txt'
34	1Timoty3.txt'	81	Acts26.txt'	128	John12.txt'
35	1Timoty4.txt'	82	Acts27.txt'	129	John13.txt'
36	1Timoty5.txt'	83	Acts28.txt'	130	John14.txt'
37	1Timoty6.txt'	84	Acts3.txt'	131	John15.txt'
38	2Coronthians1.txt'	85	Acts4.txt'	132	John16.txt'
39	2Coronthians10.txt'	86	Acts5.txt'	133	John17.txt'
40	2Coronthians11.txt'	87	Acts6.txt'	134	John18.txt'
41	2Coronthians12.txt'	88	Acts7.txt'	135	John19.txt'
42	2Coronthians13.txt'	89	Acts8.txt'	136	John2.txt'
43	2Coronthians2.txt'	90	Acts9.txt'	137	John20.txt'
44	2Coronthians3.txt'	91	colasians1.txt'	138	John21.txt'
45	2Coronthians4.txt'	92	colasians2.txt'	139	John3.txt'
46	2Coronthians5.txt'	93	colasians3.txt'	140	John4.txt'
47	2Coronthians6.txt'	94	colasians4.txt'	141	John5.txt'

Appendex IV:list of prefix and suffix

Prefix lists					Suffix lists				
የ	ስለ	የሚ	እየ	ስለሚ	ች	ኝ	ችን	ቸው	ዊት
እያ	እንደ	አል	አለ	በ	ና	ዎች	ኛ	ዎቻቸው ም	ውም
ለ	ከ	ይ	እንዳ ይ	ሲ	ው	ዎች ም	ውያን	ዎቹ	ናቸው
እንዲ	እስከ	ከነ	እን	እነ	ባቸው	ዊያን	ነት	ያዊ	ን
					ት	ሉ	ችው	ዊ	ዊነ
					ቹ ን	ዬ	ዎ	ህ	ሽ
					ዋ	ሁ	ለት	ላት	ላቸው
					ላችሁ	በት	ባት	ባቸው	ባችሁ
					ቱ	ሽ	ይቱ	የው	ኞች

Appendix V: Python Program Code

For Opening files in the root and child directory

```
def File_Lists():
    #the root directory of all the files
    directory=os.path.join("","c:\Holy Bible")
    fileNames=[]
    for root, dirs, files in os.walk(directory):
        for file in files:
            if file.endswith(".txt"):
                fileName.append(os.path.join(root,file))
    return filename
```

Checking weather a file is ASCII

```
def unicodeEncoded(filename):
    f=codecs.open(filename,'r')
    encodingCheck=f.read()
    f.close()
    try:
        encodingCheck.decode('ascii')
        return False
    except UnicodeDecodeError:
        return True
```

Collecting Unicode encoded files only

```
def unicodeFiles():
    txtFiles=File_Lists()
    for filename in txtFiles:
        if not unicodeEncoded(filename):
            txtFiles.remove(filename)
    return txtFiles
```

String Clearing using Unicode order

```
def txt_clearing(filename):
    str=codecs.open(filename,'r','utf-8').read()
    punc=codecs.open("punc.txt",'r','utf-8').read()
    filtered="".join(i for i in str if (ord(i) in range(4600,5000) or ord(i) in range(1,33)))
    for i in filtered.split():
        wordlist.append(filter(lambda x: x not in punc, i))
    return wordlist
```

Word Rooter

```
def rooter(word1):
    wordlst=[]
      sufx = next((word for word in suffix if (len(word1)>=2*len(word)+1 and
      word1.endswith(word))),False)
    pref = next((word for word in prefix if (len(word1)>=2*len(word)+1 and
    word1.startswith(word))),False)
    if pref==False: pref=""
    if sufx==False: sufx=""
                word=word1[len(pref):len(word1)-len(sufx)]
    if len(word)<=2:    word=word1
    for i in range(0,7):
        wordlst.append(word[:len(word)-1]    +    unichr((4608+((ord(word[len(word)-1])-
4608)/8)*8)+i))
    item=wordlst+[prf+i for prf in prefix for i in wordlst]
    return item
```

The regular expression to check Semantically related terms

```
def related(chklst,lists):
    rel=[]
    for mystring in lists:
        if any(re.match(regex,mystring,re.U) for regex in rooter(chklst)):
            rel.append(mystring)
        toCompare=[x for x in  tocheck if x not in rel]
```

Posting file Maker

```
group=pickle.load(open("TotalList"))
```

```
documents=pickle.load(open("filenames.txt"))
PostingFile=codecs.open("PostFile",'w')
Tfile=[]
for i in documents:
   InvFile=[]
   doc=codecs.open(i,'r','utf-8').read()
   for k in group:
      pos=[]
      for p in k:
         pos+=[m.start() for m in re.finditer(p, doc)]
      InvFile.append(pos)
   Tfile.append(InvFile)
pickle.dump(Tfile,PostingFile)
return rel
```

Unique word generator from all word lists

```
Def Unique(wordlist wrdLst):
        NwWrdList=[]
        For all I in wrdLst:
                If I not in NwWrdList:
                        NwWrdList.append(I)
                Else:
                        break
        Return NwWrdList
```

Appendix VI: Multi level Indexing

434	ሄደ	ሄደ	ሄደ
435	ሄደሃል	ሄደ	ሄደሃል
436	ሄደሃልና	ሄደ	ሄደሃልና
437	ሄደሀ	ሄደ	ሄደሀ
438	ሄደሀም	ሄደ	ሄደሀም
439	ሄደለት	ሄደ	~~ሄደለት~~
440	ሄደም	ሄደ	ሄደም
441	ሄደሻል	ሄደ	ሄደሻል
442	ሄደሽ	ሄደ	ሄደሽ
443	ሄደች	ሄደ	ሄደች
444	ሄደችም	ሄደ	ሄደችም
445	ሄደና	ሄደ	ሄደና
446	ሄደን	ሄደ	ሄደን
447	ሄደእንዲህም	ሄደ	ሄደእንዲህም
448	ሄደዋል	ሄደ	ሄደዋል
449	ሄደዋልና	ሄደ	ሄደዋልና
450	ሄደው	ሄደ	ሄደው
451	ሄደውማል	ሄደ	ሄደውማል
452	ሄደውም	ሄደ	ሄደውም
453	ሄዱ	ሄደ	ሄዱ
454	ሄዱም	ሄደ	ሄዱም
455	ሄዱብኝ	ሄደ	ሄዱብኝ
456	ሄዱና	ሄደ	ሄዱና
457	ሄዳ	ሄደ	ሄዳ
458	ሄዳም	ሄደ	ሄዳም
459	ሄዳችሁ	ሄደ	ሄዳችሁ
460	ሄዳችሁም	ሄደ	ሄዳችሁም
461	ሄዳችኋል	ሄደ	ሄዳችኋል
462	ሄድሁ	ሄደ	ሄድሁ
463	ሄድሁባቸው	ሄደ	ሄድሁባቸው
464	ሄድህ	ሄደ	ሄድህ
465	ሄድሽ	ሄደ	ሄድሽ
466	ሄድሽበት	ሄደ	~~ሄድሽበት~~
467	ሄድን	ሄደ	ሄድን
468	ሄድንና	ሄደ	ሄድንና
469	ሄዶ	ሄደ	ሄዶ
470	ሄዶም	ሄደ	ሄዶም
471	ሄዶአል	ሄደ	ሄዶአል
472	ሄዶአልና	ሄደ	ሄዶአልና
473	ለሚሄዱ	ሄደ	ለሚሄዱ
474	ለሚሄዱባትም	ሄደ	ለሚሄዱባትም
475	ለሚሄዱትም	ሄደ	ለሚሄዱትም
476	ለሚሄድ	ሄደ	ለሚሄድ
477	ሊሄዱ	ሄደ	ሊሄዱ
478	ሊሄድ	ሄደ	ሊሄድ
479	ሊሄድበት	ሄደ	~~ሊሄድበት~~
480	ሊሄድና	ሄደ	ሊሄድና
481	ስለሄዱ	ሄደ	ስለሄዱ
482	በሄደ	ሄደ	በሄደ
483	በሄደም	ሄደ	በሄደም
484	በሄደበት	ሄደ	~~በሄደበት~~
485	በሄደች	ሄደ	በሄደች
486	በሄዱ	ሄደ	በሄዱ
487	በሄዱት	ሄደ	~~በሄዱት~~
488	በሄዳችሁ	ሄደ	በሄዳችሁ
489	በሄዳችሁበት	ሄደ	~~በሄዳችሁበት~~
549	ሄደሃል	ሄደሀ	ሄደሃል
550	ሄደሃልና	ሄደሀ	ሄደሃልና
551	ሄደሀ	ሄደሀ	ሄደሀ
552	ሄደሀም	ሄደሀ	ሄደሀም
553	ሄደለት	ሄደለተ	~~ሄደለት~~
554	ሄደም	ሄደመ	ሄደም
555	በሄደም	ሄደመ	በሄደም
556	አልሄደም	ሄደመ	አልሄደም
557	አልሄደምን	ሄደመ	አልሄደምን
558	ሄደሻል	ሄደሸ	ሄደሻል
559	ሄደሽ	ሄደሸ	ሄደሽ
560	በሄደበት	ሄደበ	~~በሄደበት~~
561	በየሄደበትም	ሄደበ	በየሄደበትም
562	የሄደበትን	ሄደበ	የሄደበትን
563	ሄደች	ሄደቸ	ሄደች
564	ሄደችም	ሄደቸ	ሄደችም
565	በሄደች	ሄደቸ	በሄደች
566	ሄደእንዲህም	ሄደእንዲህ	ሄደእንዲህም
567	ሄደዋል	ሄደወ	ሄደዋል
568	ሄደዋልና	ሄደወ	ሄደዋልና
569	ሄደው	ሄደወ	ሄደው
570	ሄደውማል	ሄደወ	ሄደውማል
571	ሄደውም	ሄደወ	ሄደውም
572	የሄደው	ሄደወ	የሄደው
573	የሚሄደው	ሄደወ	የሚሄደው
574	የሚሄደውም	ሄደወ	የሚሄደውም
575	የሚሄደውን	ሄደወ	የሚሄደውን
576	የሚሄደውንም	ሄደወ	የሚሄደውንም
577	የሚሄደውንና	ሄደወ	የሚሄደውንና
578	ሄዱም	ሄዱመ	ሄዱም
579	አልሄዱም	ሄዱመ	አልሄዱም
580	አልሄዱምና	ሄዱመ	አልሄዱምና
581	አልሄዱምን	ሄዱመ	አልሄዱምን
582	አትሄዱም	ሄዱመ	አትሄዱም
583	አይሄዱም	ሄዱመ	አይሄዱም
584	አይሄዱምም	ሄዱመ	አይሄዱምም
585	አይሄዱምና	ሄዱመ	አይሄዱምና
586	ከሄዱም	ሄዱመ	ከሄዱም
587	ሄዱብኝ	ሄዱበ	ሄዱብኝ
588	ለሚሄዱባትም	ሄዱበ	ለሚሄዱባትም
589	ከሄዱባቸው	ሄዱበ	ከሄዱባቸው
590	የሚሄዱበት	ሄዱበ	~~የሚሄዱበት~~
591	የሚሄዱበትን	ሄዱበ	የሚሄዱበትን
592	የሚሄዱበትንም	ሄዱበ	የሚሄዱበትንም
593	ለሚሄዱትም	ሄዱተ	ለሚሄዱትም
594	በሄዱት	ሄዱተ	~~በሄዱት~~
595	ከሄዱትም	ሄዱተ	ከሄዱትም
596	የሄዱት	ሄዱተ	~~የሄዱት~~
597	የሄዱትን	ሄዱተ	የሄዱትን
598	የሚሄዱት	ሄዱተ	~~የሚሄዱት~~
599	የሚሄዱትን	ሄዱተ	የሚሄዱትን
600	ሄዳችሁ	ሄዳችሀ	ሄዳችሁ
601	ሄዳችሁም	ሄዳችሀ	ሄዳችሁም
602	በሄዳችሁ	ሄዳችሀ	በሄዳችሁ
603	በሄዳችሁበት	ሄዳችሀ	~~በሄዳችሁበት~~
604	አልሄዳችሁምና	ሄዳችሀ	አልሄዳችሁምና

490	በሄድሁ	ሄደ	በሄድሁ
491	በሄድሁበትም	ሄደ	በሄድሁበትም
492	በሄድህ	ሄደ	በሄድህ
493	በሄድህበት	ሄደ	~~በሄድህበት~~
494	በሄድህበትም	ሄደ	በሄድህበትም
495	በሄድንበት	ሄደ	~~በሄድንበት~~
496	በሄድንባትም	ሄደ	በሄድንባትም
497	በየሄደበትም	ሄደ	በየሄደበትም
498	አልሄደም	ሄደ	አልሄደም
499	አልሄደምን	ሄደ	አልሄደምን
500	አልሄዱም	ሄደ	አልሄዱም
501	አልሄዱምና	ሄደ	አልሄዱምና
502	አልሄዱምን	ሄደ	አልሄዱምን
503	አልሄዳችሁምና	ሄደ	አልሄዳችሁምና
504	አልሄዳችሁበትምና	ሄደ	አልሄዳችሁበትምና
505	አልሄድሁም	ሄደ	አልሄድሁም
506	አልሄድህምና	ሄደ	አልሄድህምና
507	አልሄድም	ሄደ	አልሄድም
508	አልሄድሽም	ሄደ	አልሄድሽም
509	አልሄድንምና	ሄደ	አልሄድንምና
510	አትሄዱም	ሄደ	አትሄዱም
511	አትሄድም	ሄደ	አትሄድም
512	አይሄዱ	ሄደ	አይሄዱ
513	አይሄዱም	ሄደ	አይሄዱም
514	አይሄዱምም	ሄደ	አይሄዱምም
515	አይሄዱምና	ሄደ	አይሄዱምና
516	አይሄድም	ሄደ	አይሄድም
517	ከሄደ	ሄደ	ከሄደ
518	ከሄዱ	ሄደ	ከሄዱ
519	ከሄዱም	ሄደ	ከሄዱም
520	ከሄዱባቸው	ሄደ	ከሄዱባቸው
521	ከሄዱትም	ሄደ	ከሄዱትም
522	ከሄድሁ	ሄደ	ከሄድሁ
523	ከሚሄድ	ሄደ	ከሚሄድ
524	የሄደ	ሄደ	የሄደ
525	የሄደበትን	ሄደ	የሄደበትን
526	የሄደው	ሄደ	የሄደው
527	የሄዱ	ሄደ	የሄዱ
528	የሄዱት	ሄደ	~~የሄዱት~~
529	የሄዱትን	ሄደ	የሄዱትን
530	የሄድህ	ሄደ	የሄድህ
531	የሚሄደው	ሄደ	የሚሄደው
532	የሚሄደውም	ሄደ	የሚሄደውም
533	የሚሄደውን	ሄደ	የሚሄደውን
534	የሚሄደውንም	ሄደ	የሚሄደውንም
535	የሚሄደውንና	ሄደ	የሚሄደውንና
536	የሚሄዱ	ሄደ	የሚሄዱ
537	የሚሄዱበት	ሄደ	~~የሚሄዱበት~~
538	የሚሄዱበትን	ሄደ	የሚሄዱበትን
539	የሚሄዱበትንም	ሄደ	የሚሄዱበትንም
540	የሚሄዱት	ሄደ	~~የሚሄዱት~~
541	የሚሄዱትን	ሄደ	የሚሄዱትን
542	የሚሄድ	ሄደ	የሚሄድ
543	የሚሄድበት	ሄደ	~~የሚሄድበት~~
544	የሚሄድበትም	ሄደ	የሚሄድበትም
545	የሚሄድበትን	ሄደ	የሚሄድበትን
546	የሚሄድበትንም	ሄደ	የሚሄድበትንም
547	የሚሄድባትም	ሄደ	የሚሄድባትም

605	አልሄዳችሁበትምና	ሄዳችሀ	አልሄዳችሁበትምና
606	ሄዳችኋል	ሄዳችኋለ	ሄዳችኋል
607	ሄድሁ	ሄድሀ	ሄድሁ
608	ሄድሁባቸው	ሄድሀ	ሄድሁባቸው
609	ሄድህ	ሄድሀ	ሄድህ
610	በሄድሁ	ሄድሀ	በሄድሁ
611	በሄድሁበትም	ሄድሀ	በሄድሁበትም
612	በሄድህ	ሄድሀ	በሄድህ
613	በሄድህበት	ሄድሀ	~~በሄድህበት~~
614	በሄድህበትም	ሄድሀ	በሄድህበትም
615	አልሄድሁም	ሄድሀ	አልሄድሁም
616	አልሄድህምና	ሄድሀ	አልሄድህምና
617	ከሄድሁ	ሄድሀ	ከሄድሁ
618	የሄድህ	ሄድሀ	የሄድህ
619	ሄድሽ	ሄድሽ	ሄድሽ
620	ሄድሽበት	ሄድሽ	~~ሄድሽበት~~
621	አልሄድሽም	ሄድሽ	አልሄድሽም
622	ሊሄድበት	ሄድበ	~~ሊሄድበት~~
623	የሚሄድበት	ሄድበ	~~የሚሄድበት~~
624	የሚሄድበትም	ሄድበ	የሚሄድበትም
625	የሚሄድበትን	ሄድበ	የሚሄድበትን
626	የሚሄድበትንም	ሄድበ	የሚሄድበትንም
627	የሚሄድባትም	ሄድበ	የሚሄድባትም
628	ሄድን	ሄድነ	ሄድን
629	ሄድንና	ሄድነ	ሄድንና
630	ሊሄድና	ሄድነ	ሊሄድና
631	በሄድንበት	ሄድነ	~~በሄድንበት~~
632	በሄድንባትም	ሄድነ	በሄድንባትም
633	አልሄድንምና	ሄድነ	አልሄድንምና
634	የሚሄድንም	ሄድነ	የሚሄድንም
635	ሄዶአል	ሄዶአለ	ሄዶአል
636	ሄዶአልና	ሄዶአለ	ሄዶአልና
637	ሄጃለሁ	ሄጀ	ሄጃለሁ
638	ሄጃለሁና	ሄጀ	ሄጃለሁና
639	ሄጇ	ሄጀ	ሄጇ
640	ሄጇም	ሄጀ	ሄጇም
641	ሄጇስ	ሄጀ	ሄጇስ
642	ሄጃለሁ	ሄጃለሀ	ሄጃለሁ
643	ሄጃለሁና	ሄጃለሀ	ሄጃለሁና

Appendix VII: Appendix VI: Sample words Preprocessed with LA

Word	CV	Word	CV
በሀያ	ሀየ	ላክሁባቸው	ላክሀ
በሀያኛው	ሀየ	ላክሁት	ላክሀ
በሀያኛውም	ሀየ	ለላክሃቸው	ላክሀ
በሀያው	ሀየ	አልላክሁህም	ላክሀ
ከሀያ	ሀየ	አልላክሁበሀምን	ላክሀ
ከሀያኛው	ሀየ	የላክሁላችሁን	ላክሀ
ከሀያውም	ሀየ	የላክሁትንም	ላክሀ
የሀያ	ሀየ	የላክህበኝ	ላክሀ
ሀያው	ሀያወ	የላክህበኝን	ላክሀ
ሀያውን	ሀያወ	ላክሽ	ላክሸ
በሀያው	ሀያወ	ላክኋቸው	ላክኁ
ከሀያውም	ሀያወ	በላክኋችሁ	ላክኁ
ሀያዘጠኝ	ሀያዘጠኘ	አልላክኋቸውም	ላክኁ
ሁለመናውን	ሁለ	አልላክኋቸውምና	ላክኁ
ሁለተኛ	ሁለ	ላክነው	ላክነ
ሁለተኛም	ሁለ	የላክናቸውን	ላክነ
ሁለተኛው	ሁለ	ላክኸን	ላክኸ
ሁለተኛውም	ሁለ	ላክኸኝ	ላክኸ
ሁለተኛውን	ሁለ	የላክኸው	ላክኸ
ሁለተኛውንም	ሁለ	የላክኸውንም	ላክኸ
ሁለተኛውንና	ሁለ	አትላወስ	ላወሰ
ሁለተኛይቱ	ሁለ	አትላወሺ	ላወሸ
ሁለተኛይቱም	ሁለ	እንዳላወቁ	ላወቀ
ሁለተኛይቱን	ሁለ	ላወጣላት	ላወጣላተ
ሁለቱ	ሁለ	እንዳላውቅ	ላውቀ
Word	CV	Word	CV
ሁለቱም	ሁለ	የማላውቀው	ላውቀ
ሁለቱሳ	ሁለ	የማላውቀውንም	ላውቀ
ሁለቱን	ሁለ	የማላውቅ	ላውቀ
ሁለቱንም	ሁለ	ከላውዴዎን	ላውዴወ
ሁለታቸው	ሁለ	ከላውዴዎንም	ላውዴወ
ሁለታቸውም	ሁለ	ላውጣ	ላውጠ
ሁለታቸውን	ሁለ	ላውጣላችሁ	ላውጠ
ሁለታችሁን	ሁለ	ላዘዘው	ላዘዘ
ሁለታችን	ሁለ	ላዘዝሁት	ላዘዘ
ሁለት	ሁለ	ላዘዝኋችሁ	ላዘዘ
ሁለትም	ሁለ	ላዙርህን	ላዙርህ
ሁለትን	ሁለ	ላየው	ላየወ
ሁለንተናህ	ሁለ	ላየውም	ላየወ
ሁለንተናህን	ሁለ	የማላየውን	ላየወ
ሁለንተናሽ	ሁለ	ላያችሁ	ላያቸ
ሁለንተናውም	ሁለ	በላያችሁም	ላያቸ
ሁለንተናውን	ሁለ	በላያችሁና	ላያቸ
ሁሉ	ሁለ	በላያችን	ላያቸ
ሁሉም	ሁለ	በላያችንም	ላያቸ
ሁሉስ	ሁለ	በላያችንና	ላያቸ
ሁሉና	ሁለ	ከላያቸው	ላያቸ
ሁሉን	ሁለ	ላይሳ	ላይሰ
ሁሉንም	ሁለ	ላይስ	ላይሰ
ሁሉንስ	ሁለ	በላይሽ	ላይሸ
ሁላ	ሁለ	ላይኛው	ላይኛወ
Word	CV	Word	CV
ሁላሁሉ	ሁለ	ላይኛውን	ላይኛወ
ሁላቸው	ሁለ	ላይኛውንና	ላይኛወ
ሁላቸውም	ሁለ	በላይኛው	ላይኛወ
ሁላቸውንም	ሁለ	በላይኛውም	ላይኛወ
ሁላችሁ	ሁለ	ከላይኛው	ላይኛወ
ሁላችሁም	ሁለ	የላይኛውን	ላይኛወ
ሁላችሁን	ሁለ	የላይኛውንና	ላይኛወ
ሁላችሁንስ	ሁለ	ላይኛይቱ	ላይኛይተ
ሁላችን	ሁለ	በላይኛይቱ	ላይኛይተ
ሁላችንም	ሁለ	ላይኞቹ	ላይኞቸ
ሁላችንን	ሁለ	በላይዋ	ላይወ
ሁል	ሁለ	በላይዋም	ላይወ
ሁልጊዜ	ሁለ	በላይዋስ	ላይወ
ሁልጊዜም	ሁለ	ላይደለ	ላይደለ
ሁልጊዜስ	ሁለ	ላደረገ	ላደረገ
ለሁለተኛው	ሁለ	ላደረገች	ላደረገ
ለሁለተኛይቱም	ሁለ	ላደረገው	ላደረገ
ለሁለቱ	ሁለ	ላደረጉ	ላደረገ
ለሁለቱም	ሁለ	ላደረጉበኝም	ላደረገ
ለሁለት	ሁለ	ላደረጉት	ላደረገ
ለሁሉ	ሁለ	ላደረጋት	ላደረገ
ለሁሉም	ሁለ	እንዳላደረገ	ላደረገ
ለሁላችሁ	ሁለ	እንዳላደረግሁባት	ላደረገ
ለሁላችን	ሁለ	ላደሩበት	ላደሩበተ
ለሁላችንም	ሁለ	ላደርገው	ላደርገ
Word	CV	Word	CV
በሁለተኛ	ሁለ	ላደርጋት	ላደርገ
በሁለተኛው	ሁለ	ላደርግ	ላደርገ
በሁለተኛውም	ሁለ	ላደርግላችሁ	ላደርገ
በሁለተኛውና	ሁለ	ላደርግልህ	ላደርገ
በሁለተኛይቱ	ሁለ	እንዳላደርጋት	ላደርገ
በሁለቱ	ሁለ	እንዳላደርግ	ላደርገ

በሁለቱም	ሁለ	እንዳላደርግሽ	ላደርገ
በሁለታቸው	ሁለ	እንዳላደርግባችሁ	ላደርገ
በሁለታችን	ሁለ	እንዳላደርግብሽ	ላደርገ
በሁለት	ሁለ	እንዳላዳናችሁኝም	ላዳናችሁኘ
በሁለትም	ሁለ	ላድርገው	ላድርገ
በሁለትና	ሁለ	ላድርጋቸውን	ላድርገ
በሁለንተናዋ	ሁለ	ላድርግ	ላድርገ
በሁሉ	ሁለ	ላድርግላችሁ	ላድርገ
በሁሉም	ሁለ	ላድርግልህ	ላድርገ
በሁላቸው	ሁለ	ላድርግልሽ	ላድርገ
በሁላቸውም	ሁለ	ላድናቸው	ላድነ
በሁላችሁ	ሁለ	ላድን	ላድነ
በየሁለቱም	ሁለ	እንዳላገለገሉ	ላገለገ
በየሁለትም	ሁለ	ላገባት	ላገባተ
ከሁለተኛው	ሁለ	ላገኘው	ላገኘ
ከሁለተኛውም	ሁለ	ላገኙ	ላገኘ
ከሁለቱ	ሁለ	እንዳላገኘሁበት	ላገኘ
ከሁለቱም	ሁለ	እንዳላገኙ	ላገኘ
ከሁለታቸው	ሁለ	እንዳላገኛችሁ	ላገኘ
ከሁለታችንም	ሁለ	ከሚላገድ	ላገደ
Word	**CV**	**Word**	**CV**
ከሁለት	ሁለ	ላግኝ	ላግኘ
ከሁለትም	ሁለ	ላጠምዳችሁ	ላጠምደ
ከሁሉ	ሁለ	ላጠምድ	ላጠምደ
ከሁሉም	ሁለ	ላጠረው	ላጠረ
ከሁላቸው	ሁለ	ላጠበን	ላጠበ
ከሁላችሁ	ሁለ	እንዳላጠፉ	ላጠፈ
ከሁላችሁም	ሁለ	እንዳላጠፋ	ላጠፈ
ከሁላችን	ሁለ	እንዳላጠፋህ	ላጠፈ

የሁለተኛው	ሁለ	እንዳላጠፋህም	ላጠፈ
የሁለተኛውም	ሁለ	እንዳላጠፋት	ላጠፈ
የሁለተኛውንም	ሁለ	እንዳላጠፋችሁ	ላጠፈ
የሁለተኛይቱ	ሁለ	ላጣብቀው	ላጣብቀ
የሁለተኛይቱም	ሁለ	እንዳላጣብቅህ	ላጣብቀ
የሁለተኛይቱን	ሁለ	የላጣቸውንም	ላጣቸውነ
የሁለተኛይቱንም	ሁለ	ላጨ	ላጨ
የሁለቱ	ሁለ	ላጨው	ላጨ
የሁለቱም	ሁለ	ላጭተው	ላጨ
የሁለቱን	ሁለ	አትላጩ	ላጨ
የሁለቱንም	ሁለ	አይላጩ	ላጨ
የሁለት	ሁለ	አይላጩላቸውም	ላጨ
የሁሉ	ሁለ	አይላጭም	ላጨ
የሁሉም	ሁለ	ከላጩት	ላጨ
የሁሉን	ሁለ	የሚላጩትን	ላጨ
የሁሉንም	ሁለ	አይላጩላቸውም	ላጩላቸ
የሁላቸው	ሁለ	ከላጩት	ላጩተ
የሁላችሁ	ሁለ	የሚላጩትን	ላጩተ
Word	**CV**	**Word**	**CV**
የሁላችሁም	ሁለ	ላጭተው	ላጭተ
የሁላችሁን	ሁለ	ላፈረሱት	ላፈረሱተ
የሁላችን	ሁለ	እንዳላፈሰስህ	ላፈሰሰ
የሁላችንን	ሁለ	እንዳላፍር	ላፍረ
ሁለመናውን	ሁለመናወ	እንዳላፍርም	ላፍረ
ሁለመናውንም	ሁለመናወ	ሌለ	ሌለ
ሁለተኛ	ሁለተ	ሌለበት	ሌለ
ሁለተኛም	ሁለተ	ሌለኝ	ሌለ
ሁለተኛው	ሁለተ	ሌለው	ሌለ
ሁለተኛውም	ሁለተ	ሌሊቱ	ሌለ

ሁለተኛውን	ሁለተ	ሌሊቱም	ሌለ
ሁለተኛውንም	ሁለተ	ሌሊቱን	ሌለ
ሁለተኛውንና	ሁለተ	ሌሊቱንም	ሌለ
ሁለተኛይቱ	ሁለተ	ሌሊቱንና	ሌለ
ሁለተኛይቱም	ሁለተ	ሌሊት	ሌለ
ሁለተኛይቱን	ሁለተ	ሌሊትም	ሌለ
ሁለቱ	ሁለተ	ሌሊትስ	ሌለ
ሁለቱም	ሁለተ	ሌሊትና	ሌለ
ሁለቱሳ	ሁለተ	ሌሊትን	ሌለ
ሁለቱን	ሁለተ	ሌላ	ሌለ
ሁለቱንም	ሁለተ	ሌላም	ሌለ
ሁለታቸው	ሁለተ	ሌላስ	ሌለ
ሁለታቸውም	ሁለተ	ሌላቸው	ሌለ
ሁለታቸውን	ሁለተ	ሌላችሁ	ሌለ
ሁለታችሁን	ሁለተ	ሌላን	ሌለ
ሁለታችን	ሁለተ	ሌላንም	ሌለ
Word	**CV**	**Word**	**CV**
ሁለት	ሁለተ	ሌላው	ሌለ
ሁለትም	ሁለተ	ሌላውም	ሌለ
ሁለትን	ሁለተ	ሌላውን	ሌለ
ለሁለተኛው	ሁለተ	ሌላውንም	ሌለ
ለሁለተኛይቱም	ሁለተ	ሌላዪቱን	ሌለ
ለሁለቱ	ሁለተ	ሌላይቱ	ሌለ
ለሁለቱም	ሁለተ	ሌላይቱም	ሌለ
ለሁለት	ሁለተ	ሌላይቱንም	ሌለ
በሁለተኛ	ሁለተ	ሌሎቹ	ሌለ
በሁለተኛው	ሁለተ	ሌሎቹም	ሌለ
በሁለተኛውም	ሁለተ	ሌሎቹን	ሌለ
በሁለተኛውና	ሁለተ	ሌሎች	ሌለ

በሁለተኛይቱ	ሁለተ	ሌሎችም	ሌለ
በሁለቱ	ሁለተ	ሌሎችን	ሌለ
በሁለቱም	ሁለተ	ሌሎችንም	ሌለ
በሁሊታቸው	ሁለተ	ለሌለበት	ሌለ
በሁሊታችን	ሁለተ	ለሌለው	ሌለ
በሁለት	ሁለተ	ለሌለውም	ሌለ
በሁለትም	ሁለተ	ለሌለውና	ሌለ
በሁለትና	ሁለተ	ለሌሊት	ሌለ
በየሁለቱም	ሁለተ	ለሌላ	ሌለ
በየሁለትም	ሁለተ	ለሌላቸው	ሌለ
ከሁለተኛው	ሁለተ	ለሌላቸውና	ሌለ
ከሁለተኛውም	ሁለተ	ለሌላው	ሌለ
ከሁለቱ	ሁለተ	ለሌላውም	ሌለ
ከሁለቱም	ሁለተ	ለሌላይቱ	ሌለ
Word	**CV**	**Word**	**CV**
ከሁሊታቸው	ሁለተ	ለሌሎቹ	ሌለ
ከሁሊታችንም	ሁለተ	ለሌሎች	ሌለ
ከሁለት	ሁለተ	ለሌሎችም	ሌለ
ከሁለትም	ሁለተ	ስለሌለ	ሌለ
የሁለተኛው	ሁለተ	ስለሌለሽ	ሌለ
የሁለተኛውም	ሁለተ	ስለሌለን	ሌለ
የሁለተኛውንም	ሁለተ	ስለሌለኝ	ሌለ
የሁለተኛይቱ	ሁለተ	ስለሌለው	ሌለ

የሁለተኛይቱም	ሁለተ	ስለሌላቸው	ሌለ
የሁለተኛይቱን	ሁለተ	ስለሌላቸውም	ሌለ
የሁለተኛይቱንም	ሁለተ	ስለሌላችሁ	ሌለ
የሁለቱ	ሁለተ	በሌለ	ሌለ
የሁለቱም	ሁለተ	በሌለበት	ሌለ
የሁለቱን	ሁለተ	በሌለበትም	ሌለ
የሁለቱንም	ሁለተ	በሌለባት	ሌለ
የሁለት	ሁለተ	በሌለባቸው	ሌለ
ሁለንተናህ	ሁለንተነ	በሌለው	ሌለ
ሁለንተናህን	ሁለንተነ	በሌሊት	ሌለ
ሁለንተናሽ	ሁለንተነ	በሌሊትም	ሌለ
ሁለንተናችሁን	ሁለንተነ	በሌሊትና	ሌለ
ሁለንተናዋም	ሁለንተነ	በሌላ	ሌለ
ሁለንተናው	ሁለንተነ	በሌላም	ሌለ
ሁለንተናውም	ሁለንተነ	በሌላት	ሌለ
ሁለንተናውን	ሁለንተነ	በሌላው	ሌለ
በሁለንተናዋ	ሁለንተነ	በሌላውም	ሌለ
ሁሉስ	ሁሉሰ	በሌሎቹ	ሌለ
Word	**CV**	**Word**	**CV**
ሁሉና	ሁሉነ	በሌሎች	ሌለ
ሁሉን	ሁሉነ	በሌሎችም	ሌለ
ሁሉንም	ሁሉነ	ከሌለ	ሌለ
ሁሉንስ	ሁሉነ	ከሌለበት	ሌለ

የሁሉን	ሁሉነ	ከሌለባቸው	ሌለ
የሁሉንም	ሁሉነ	ከሌለች	ሌለ
ሁላሁሉ	ሁላሀ	ከሌለኝ	ሌለ
ሁላቸው	ሁላቸ	ከሌለው	ሌለ
ሁላቸውም	ሁላቸ	ከሌለውም	ሌለ
ሁላቸውንም	ሁላቸ	ከሌሊቱ	ሌለ
ሁላችሁ	ሁላቸ	ከሌሊቱም	ሌለ
ሁላችሁም	ሁላቸ	ከሌሊት	ሌለ
ሁላችሁን	ሁላቸ	ከሌላ	ሌለ
ሁላችሁንስ	ሁላቸ	ከሌላም	ሌለ
ሁላችን	ሁላቸ	ከሌላቸው	ሌለ
ሁላችንም	ሁላቸ	ከሌላው	ሌለ
ሁላችንን	ሁላቸ	ከሌላዪቱም	ሌለ
ለሁላችሁ	ሁላቸ	ከሌላይቱ	ሌለ
ለሁላችን	ሁላቸ	ከሌላይቱም	ሌለ
ለሁላችንም	ሁላቸ	ከሌሎቹ	ሌለ
በሁላቸው	ሁላቸ	ከሌሎቹም	ሌለ
በሁላቸውም	ሁላቸ	ከሌሎች	ሌለ
በሁላችሁ	ሁላቸ	ከሌሎችም	ሌለ
ከሁላቸው	ሁላቸ	የሌለ	ሌለ
ከሁላችሁ	ሁላቸ	የሌለበት	ሌለ
ከሁላችሁም	ሁላቸ	የሌለበትን	ሌለ

Certification

I certify that this research does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any university; and that to the best of my knowledge and belief it does not contain any material previously published or written by another person except where due reference is made in the text.