Jimma University

Jimma Institute of Technology

School of Graduate Studies

Department of information Technology

Applications of Information Retrieval for Afaan Oromo text based on Semantic-based Indexing

By

Berhanu Anbase

Jimma University

May, 2019

Jimma University

Jimma Institute of Technology

School of Graduate Studies

Department of information Technology

Applications of Information Retrieval for Afaan Oromo text based on Semantic-based Indexing

By

Berhanu Anbase Bedada

A Thesis submitted to Faculty of computing of Jimma University in partial fulfillment of the Requirement for the Degree of Masters of Science in Information Technology

Jimma University

May, 2019

Jimma University

Jimma Institute of Technology

School of Graduate Studies

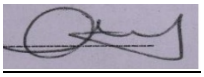
Department of information Technology

Applications of Information Retrieval for Afaan Oromo text based on Semantic-based Indexing

By

Berhanu Anbase Bedada

Name and signature of Members of the Examining Board:

Name	Title	Signature	Date
Debela Tesfaye (Ass. Professor),	Advisor		_____
Teferi Kebebew (MSc),	Co-Advisor	_____	_____
_____	Examiner	_____	_____
_____	Examiner	_____	_____

Jimma University

May, 2019

Dedication

This paper is dedicated to my father Anbase Bedada, my mother Alami Lamecha, all my brothers and sisters who were able to reap the fruit of their own. I love you all. Especially, my dedication also goes to the memory of my late sister Genet Anbase who passed away about a month and a half ago.

Geniyee you will always remain in our hearts.

Declaration

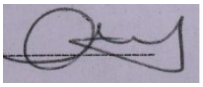
This study is my original work and has not been submitted as a partial requirement for a degree in any University and that all sources of material used for the study have been duly acknowledged.

_____

Berhanu Anbase Bedada

May, 2019

This thesis is has been submitted for examination with my approval as University Advisor.



_____

Debela Tesfaye (Ass. Professor)

May, 2019

This thesis is has been submitted for examination with my approval as University Co-Advisor.

_____

Teferi Kebebew (MSc)

May, 2019

Acknowledgment

First and foremost, I would like to thank God who helped me to succeed in all my life long learns. Next, I would like to thank my Advisor Dr Debela Tesfaye (Assistant Professor) for his valuable assistance in providing his genuine, professional advice and encouragement goes even beyond the accomplishment of this study. Some noteworthy aspects of this study are entirely because of him; all faults are in spite of him. He initiated me to do by giving precious comments on necessary points. My thanks go to him again, since it is difficult to mention his contribution to my achievements in words, it is better to say my heart has recorded it forever. And also I would like to acknowledge my co-advisor Teferi Kebebew (MSc) for all his guidance at every step of the thesis work, for patiently listening to me even during uncountable hours, for advising me to do my study, for imparting so much valuable knowledge and for all his encouragement and words of kindness. My thanks go to him again, because it is difficult to mention his contribution to my achievements in words, it is better to say my heart has recorded it forever. Next, I would like to thanks Dr Million Meshesha for his valuable assistance in providing his genuine and professional advice for reshaping my title during the title selection. Next, Words cannot express my deepest thanks to my Father Mr. Anbase Bedada and My Mother Alami Lamecha, especially she is the one who sincerely raised me with her caring and gently love and my brothers Hailemariyam Anbase, Adugna Anbase and Habtamu Anbase for their moral and financial support I received from them throughout my life of learning. My honest thanks equally go to my sisters Dayite Anbase, Genet Anbase and Wareqinesh Anbase for their help in providing me concerning background information. I would like to thanks to Assosa University for granting me to attend my post graduate study at Jimma University and all my friends who have helped me on my progress; especially Zerihun Olana, Anduwalem chekole, Workineh Tesema, Charu Haile, Tesfaye Tadale, Solomon Aregaw, Zakariyas Ayinalem and my lovely friends (Muliyee (Gizeworqi)) for their moral.

Last but not least, I would like to give my thanks to my friend for their moral support, endless love and encouragement during my study. Especially Asayehegn G/medin, you are the reason for the person I become today. I thank you all.

Table Content

Table of Contents **Page**

List of Tables

List of Tables pages

List of Figures

List of Figures pages

List of Acronyms

IR: Information Retrieval

LSI: Latent semantic indexing

SVD: Singular value decomposition

JAMA: JAva MAtrix

ORTO: Oromia Radio and Television Organization (ORTO)

VOA: Voice of America

VSM: Vector Space Model

TF: Term frequency

IDF: Inverse document frequency

TFIDF: Term frequency Inverse document frequency

Abstract

As storage becomes more sufficient and less expensive, the amount of information retained by businesses and institutions is likely to increase. For these increasing amounts of information, they need efficient and effective index structure when we have to find needed information. Indexing is an offline process for organizing collection of documents using indexing structure such as Inverted file, sequential files and signature file to save storage memory space and speed up searching time. However, since there are usually many ways to express the same concept, the terms in the user's query may not appear in a relevant document. Alternatively, many words can have more than one meaning. So the standard information retrieval models (e.g., Boolean, standard vector, probabilistic) treat words as if they are independent, although it is quite obvious that they are not. Due to these facts term matching methods are likely to miss relevant documents and also retrieve irrelevant ones. Therefore, to solve this issue, we have developed Applications of Information Retrieval for Afaan Oromo text based on Semantic-based Indexing. This technique of information retrieval can partially handle these problems by organizing terms and documents into a "semantic-based indexing" structure more appropriate for information retrieval. Semantic indexing information retrieval is a new technique which is based on vector representation of documents and user's query. Because it uses concepts or topics instead of each words to index and retrieve the documents, consequently allowing a relevant document to be retrieved even when it shares no common words with in the query.

In this study, the potential of applications of Latent Semantic indexing approach in Afaan Oromoo text retrieval is investigated. 70 Afaan Oromoo documents and nine (9) queries were used to test the approach. Automatic indexing of the documents resulted in some unique terms which are not in the stop-word list used for the thesis. Singular value decomposition (SVD) of the term by document matrix is used for indexing and retrieval system.

The performance of the system after User Relevance Feedback is measured using recall and precision. Therefore; the experiment shows that the performance of the prototype is on the average 0.67(67%) precision and 0.63(63%) recall registered. Finally, the recommendations can be forwarded based on the applications resource that the performance of the LSI system can be improved by designing good stemmer and standard corpus for Afaan Oromoo languages.

Keywords: **Information Retrieval, Afaan Oromo, Latent Semantic indexing, Singular Value Decomposition.**

Chapter One
Introduction

1.1 Background

As storage becomes more ample and less expensive, the amount of information retained by businesses and institutions is likely to increase. For these increasing amounts of information, they need efficient and effective index structure when we have to find needed information. Indexing is an offline process, that is, the end user of IR system is not directly involved of representing and organizing large document collection using indexing structure such as Inverted file, sequential files and signature file to save storage memory space and speed up searching time [15]. Therefore, most techniques to retrieving textual materials from databases/document collections (documents in an organization) depend on exact term match between terms in user's query and terms by which documents are indexed. However, since there are usually many ways to express the same concept, the terms in the user's query may not appear in a relevant document. Alternatively, many words can have more than one meaning. So the standard information retrieval models (e.g., Boolean, standard vector, probabilistic) treat words as if they are independent, although it is quite obvious that they are not[1][3]. Due to these facts term matching methods are likely to miss relevant documents and also retrieve irrelevant ones. For these reason, we would Applications of Information Retrieval for Afaan Oromo text based on Semantic-based Indexing. This technique of information retrieval can partially handle these problems by organizing terms and documents into a "semantic-based indexing" structure more appropriate for information retrieval. Semantic indexing information retrieval is a new technique which is based on vector representation of documents and user's query. Because it uses concepts or topics instead of individual words to index and retrieve the documents, consequently allowing a relevant document to be retrieved even when it shares no common words with in the query [1][2].

There are more than 80 languages in Ethiopia. Afaan Oromo is one of the languages with large number of speakers under Cushitic family [6]. As the Ethiopia's statistical report of 2007 [6] shows there are more than 25 million speakers of Afaan Oromo in Ethiopia and this fact shows that, the language has the largest speaker followed by Amharic language.

The objective of this study is to LSI approaches for Afaan Oromo textual document. In this case the approach is that the LSI can improve the effectiveness, because it uses concepts or topics instead of individual words to index and retrieve the documents. The motivation behind LSI is Afaan Oromo to allow the users to make sufficient use of the available technologies because many terms can have more than one meanings present in any language provide great difficulty in the use of information retrieval as terms in human language that occur in a particular context can be interpreted in more than one way depending on the context or semantic-based indexing. The inspiration for this research comes from the desire to Applications of Information Retrieval for Afaan Oromo text based on Semantic-based Indexing.

1.2. Statement of the Problem

In Ethiopia there are more than 80 languages. Afaan Oromo is one of the languages with large number of speakers under Cushitic family [6]. Afaan Oromo language uses Latin based script which is known as "Qubee" and that consists of thirty-one character of which five are vowels, twenty-six are consonants, out of which five are pair letters or diagraphs and fall together (a combination of two consonant characters such as *'ch'*). It is the official language of Oromia regional state of Ethiopia and also academic language for primary school of the region. In addition to this Afaan Oromo literature and folklore offered as a field of study in many universities located in Ethiopia and other countries. Currently journal, magazines, newspapers, news, online education and books are accessible in electronic format both on the Internet and on offline sources to mention a few. For these reason, huge amount of information being released with Afaan Oromoo language, because Afaan Oromoo is the language of education and research, language of administration and political welfares, language of ritual activities and social interaction and many medias used as Afaan Oromoo as a primary language; for instance, Oromia Television and Radio (Web news), Kallacha Oromiyaa, Bariisaa, Yeroo, Voice of America (VOA) (web news), and different academic and recreational medias to mention a few.

Nowadays there are used to describe information retrieval systems for Afaan Oromo uses mostly keyword-based and identify relevant documents or information by matching keywords. For instance, in Afaan Oromo the word **'Yaaduu', 'Herreguu'** and **'xiinxaluu',** all can be used to describe **'Think'** (from Afaan Oromo dictionary entitled, "Galmee jechoota Afaan Oromoo".). Thus, in a

literal search, only documents containing the literal query terms will be matched, causing the search results to exclude relevant documents [12]. Therefore, synonyms tend to reduce the recall of IR systems. And polysemy (words are characterized by different meaning based on the context). For instance, the word **"daakuu"** in Afaan Oromo can mean **'flour'**, or it can also mean **'swimming'**. So terms in a user's query will literally match terms in irrelevant documents [12]. The existing information retrieval systems are mostly keyword-based and identify relevant documents or information by matching keywords .Therefore, it does not tell the meanings and relationships of the terms and concepts used in information retrieval systems for Afaan Oromo .As a result, many words can have more than one meaning and the terms in the user query may not appear in the relevant document. Due to this, information retrieval systems for Afaan Oromo become a challenging task. Hence, in this research we propose a method to Applications of Information Retrieval for Afaan Oromo text based on Semantic-based Indexing. Because it addresses on the basis of concepts, not keywords; searches are not constrained by the specific words that users choose when they formulate queries. So By using statistical techniques that Developing applications of Semantic-based Indexing model can retrieve relevant documents even when those documents do not share any words with a query. For this purpose we had develop Semantic-based Indexing Information Retrieval for Afaan Oromo. Consequently, we have the following research questions that can be answered at the end of this thesis work.

- What are the linguistic features of the Afaan Oromo writing system and their suitable text operations in the corpus?
- Could semantic indexing improve the Afaan Oromoo IR performance?
- To what extent latent semantic indexing enables to design effective Afaan Oromo text retrieval?

1.3. Objective of the Study

The general and specific objectives of the study are described as below.

1.3.1 General Objective

The main objective of this study is to Applications of Information Retrieval for Afaan Oromo text based on Semantic-based Indexing.

1.3.2 Specific Objectives

In order to accomplish the aforementioned general objective, the following specific objectives are formulated.

- To review literatures on the topic area specifically, on previous works related with information retrieval system of Afaan Oromo and digesting concept of information retrieval including related concepts like indexing and searching.
- Review relevant works in other languages that were published on Applications Semantic-based Indexing systems and Singular Value Decomposition in the context of text retrieval.
- To perform basic text operation such as, tokenization, stop words removal, stemming, normalization and generate term by document matrix related to the purpose of this research.
- To implement a prototype system to test the findings of this research and evaluate the application of Afaan Oromo Semantic-based Indexing system.
- To examine the performance of the proposed prototype and forward recommendation for future research direction.

1.4. Scope and limitation of the study

The focus of this study is on designing Applications Semantic-based Indexing system that effectively Afaan Oromo text corpus. The work mainly implements Semantic-based indexer and searcher from corpus of Afaan Oromo textual documents. Other data types, such as image, video, and graphics are out of focus of the research.

To identify content bearing index terms and query terms a series of text operations such as tokenization, stop word removal, normalization and stemming and generate term by document matrix are applied. Additionally, stemming of index terms is not done; because Semantic-based Indexing system by its nature partially handles the problems that arise due to morphological variation of index terms. Of course, we have used.

1.5. Methodology of the study

In order to achieve the objective of this research, we started by studying existing of information retrieval methods and looking at various semantic based indexing development methodologies like literature reviews, Corpus/Data Source selection, Tools/Experimentation method, Evaluation Method.

1.5.1 Literature reviews

Many literature reviews will organize to get deeper understanding on information retrieval systems, particularly on semantic based indexing approach to text retrieval. A review of literature will also conducted to get adapted with the basic Afaan Oromo text features in relation to the topic and the objectives of the study. Different materials like books, journal articles, and previous related research work as well as electronic materials on the Web have been reviewed for this purpose

1.5.2 Corpus/Data Source selection

For this research study, the corpus will be collected from the official website of Oromiya National Regional State, Voice of America Radio Afaan Oromo service, Gumii Waaqeffattoota Addunyaa (GWA) Portal, International Bible Society official website, Oromiya Radio and Television Organization (ORTO) news and other Internet based sources.

Data set used is mainly, news articles, others resources from Holy Bible books and the Internet. The information contains in the text documents are easier to access in sufficient amounts because it is the major motive to use them. The text documents are used for indexing, because the entire text of the document is better representative of the content of the corpus. Besides, the small size of the news articles (in most cases half a page) supported the decision to use the entire document. Hence, we want to collect multiple documents from news based on domain. The different type of the data set helps evaluation of the system more generic and common.

See: http://www.orto.gov.et
See: http://www.gada.com
See: http://www.wikipedia/oromo.com
See: http://www.oromiyaa.com

1.5.3 Tools/Experimentation method

The development part used is Windows environment and the programming language used is Java; NetBeans 8.0.1 which run on the prepared corpus and JAMA (JAva MAtrix) package is a basic linear algebra package for Java; used for generating term document matrix and it is provides user-level classes for constructing and manipulating real, dense matrices. It is meant to provide sufficient functionality for routine or everyday problems, packaged in a way that is natural and understandable to non-experts. Java is dynamic programming language that is used in a wide variety of application domains and it provides a great set of tools for developers because it is a general-purpose and open source programming language. Moreover, it is optimized for software quality, developer productivity, program portability, and component integration. Nowadays Java is commonly used around the world for Internet scripting, systems programming, user interfaces, product customization, numeric programming, and accessible for this research.

1.5.4 Evaluation Method

This study involves Applications of Information Retrieval for Afaan Oromo text based on Semantic-based Indexing and evaluating the performance. For this work, corpus is prepared, queries are constructed and relevance judgment is made for evaluating effectiveness of the work to measure the performance of information retrieval using in recall and precision.

In information retrieval, the usual recall and precision are often used to measure the performance of information retrieval systems. Precision is fraction of retrieved documents that are relevant, and recall is fraction of relevant document that retrieved [2]. For this research too, Precision-recall curves are used to evaluate the performance of the system.

Then by using the formulated methodology, we would develop semantic based indexing of information retrieval for Afaan Oromo language. Finally, we would demonstrate the usability of semantic based indexing by using a prototype.

1.6. Significance of the study

The result will be show that the advantage of Applications of Information Retrieval for Afaan Oromo text based on Semantic-based Indexing. The technique can be apply to several problem where there is a need to retrieve Afaan Oromo documents from a large collection. Especially, to developing application of semantic based indexing model can be more effective in situations where high recall is need and text description are short. The method can also be extended to other application on the Afaan Oromo language like information filtering, and enables Afaan Oromo speakers to retrieving text documents in Afaan Oromo efficiently and effectively. This research will have significant contribution in an attempt to use semantic based indexing for cross-language retrieval in which Afaan Oromo is one of the components.

1.7. Organization of the Thesis

This thesis is organized in five chapters. The first chapter, presents out the background, statement of the problem, and the general and specific objectives of the study together with scope and limitations as well as methodology and the contribution of the study.

The Chapter two is devoted for literature review and related works. General concepts on information retrieval and on Afaan Oromoo writing system applicable to the research are discussed. Latent semantic indexing and related researches on latent semantic indexing are also reviewed.

The third chapter deals with detail description of the Major technique and methods used in this study and the proposed architecture of the system are presented. The major components involved in the architecture of the Applications of Information Retrieval for Afaan Oromo text based on Semantic-based Indexing system are also explained in detail.

Chapter four describes experimentation and evaluation of the proposed system used in the development of prototype LSI model, discusses the results obtained from the experiment.

Finally in the chapter five most important findings including faced challenges are written as a conclusion and forwards recommendations for the future work.

Chapter Two
Literature Review

2.1 Introduction

This chapter is concerned with the review of literature. It is broadly divided into three sections. The first section discusses about some general background information about the Afaan Oromo language and some typical characteristics of the language related to information retrieval.

In the second section discusses about concepts related to IR, indexing, the series of activities involved in document and query indexing process and components of IR system. Concepts like term extraction, semantics of the term and term weighting are briefly introduced. Evaluation of information retrieval system is also dealt with.

The third section discusses about research works related to the field of information retrieval for Afaan Oromo language especially LSI based retrieval and other languages are reviewed.

2.2. Overview of the Languages

Ethiopia is one of the multilingual countries. It constitutes more than 80 ethnic groups with diversified linguistic backgrounds [10]. Afaan Oromo is part of the Lowland East Cushitic group within the Cushitic family of the Afro-Asiatic phylum [11]. In this Cushitic branch of the Afro-asiatic language family Afaan Oromo is considered as one of the languages that are widely spoken and used in Ethiopia [11]. According to Tabor Wami [21] it is also the third most widely spoken language in Africa next to Arabic and Hausa languages. Afaan Oromo, although relatively widely distributed within Ethiopia and some neighboring countries like Kenya and Somalia, is one of the most resource scarce languages [9]. It is a common mother tongue that is used by Oromo people, who are the largest ethnic group in Ethiopia, which accounts to 34.5% of the total population [6].

Afaan Oromo has a very rich morphology like other African and Ethiopian languages [23].There have been attempts to make Afaan Oromoo have its own indigenous alphabet. Shaykh Bakri Saphalo has designed and introduced indigenous alphabet for Afaan Oromoo in 1956. However, because of

political reason, it was greeted with negative reaction from government officials and Shaykh Bakri was confined to Diredawa [38]. The first paper on using Latin for Afaan Oromo was published in 1972 By Dr Hayile Fida & co. Then in 1973, he used Latin to publish a book called Hirmaata Dubbii Afaan Oromoo and then in 1974 another book titled Bara Birraan Barihe. Soon after that, OLF adopted Latin for writing Afaan Oromo and in 1991 officially adopted, but on November 3, 1991 it adopted as official script of Afaan Oromo [9].

Currently, Afaan Oromo is the official language in the Oromia regional state (which is the largest region in Ethiopia). Since 1994 Afaan Oromo is a medium of instruction in primary schools and teacher training institutions of Oromia Regional State. In addition to this, it has been taught as a subject in secondary schools in Oromia Region. From 2003 onwards, universities like Jimma and Haramaya started offering BA program in Afaan Oromo and Literature. The pressing requirements for qualified researchers in Afaan Oromo Language and Literature is also increasing from time to time due to the fact that the language has become the official language in Oromia Regional State, and it also serves as language of the courts, religions, mass media, etc. For these reason; text books, references, magazines and other governmental documents are published using the language.

2.2.1. Dialects and varieties

Every language that is spoken over any significant area is spoken in somewhat different forms in different places; these are its regional dialects. Moreover, even in a single community, the language may be spoken differently by members of different social groups; these different forms are social dialects [33]. So, Afaan Oromo is a sociolinguistic language consisting four major varieties: Borana-Arsi-Guji Oromo, Eastern Oromo, Orma (Oromo in Kenya) and West Central Oromo. These four varieties depend on geographical area. Even if there is strong similarities among these four varieties, but there is also difference between them .There are many synonym words which make effectiveness of IR system lower [29].

2.2.2. Alphabets and Sounds

According to Taha, R [22], Afaan Oromo is a phonetic language, which means that it is spoken in the way it is written. Additionally characters sound the same in every word in contrast to English in which the same letter may sound differently in different words.

The alphabets of Afaan Oromo are often called “Qubee Afaan Oromoo”, alphabets of the Oromo language. Afaan Oromo uses Qubee (Latin based alphabet) that consists of thirty-one character of

which five are vowels, twenty-six are consonants, out of which five are pair letters or diagraphs and fall together (a combination of two consonant characters such as *'ch'*). The major representatives of sources of the sound in a language are the vowels and consonants. The Afaan Oromo alphabet characterized by capital and small letters as in the case of the English alphabet. In Afaan Oromo language, as in English language, vowels are sound makers and are sound by themselves. The basic alphabet in Afaan Oromo does not contain 'p', 'v' and 'z', since there is no Afaan Oromo words that are written by use of these characters. However, in writing Afaan Oromo language, they are included by considering borrowed terms from other languages like English.

A. Vowels-Dubbiftuu

Afaan Oromo basically has 10 phonemic vowels, five short and five long vowels, which are represented by the five basic letters such as a, e, i, o, u. It has the typical Eastern Cushitic set of five short and five long vowels by doubling the five vowel letters: 'aa', 'ee', 'ii', 'oo', 'uu' [11]. These vowels (Table1) can be divided into different categories depending how they are formulated: Front/back indicates the position of tongue that give sound in mouth, while High/Low is place of the tongue with respect the palate (the top part of the inside of the mouth) during articulation [32].

	Front	**Central**	**Back**
High	i,ii	u,uu	
Mid	e , ee	o,oo	
Low	A	aa	

Table.2.1. Afaan Oromo vowels (Dubbiistoota)

B. Consonants (dubbifamtoota)

All Afaan Oromo consonants do not differ greatly from English, but there are some exceptions and few special combination consonants letters: **ch, dh, ny, ph**, and **sh**. The combined consonant letters are known as "**Qubee dacha**". It has double consonant combinations if the technical term for stressed

words is germination. For instance, **d** used in **badaa** means ‘bad’ and **baddaa** means ‘highland’. Failure to make this distinction results in miscommunication. For examples, the word “*Qoricha*”, which is to mean ‘Medicine’, is different from “*Qorricha*” which is “the cold weather”.

		Bilabial/ Labiodental	Alveolar/ Retroflex	Palato-alveolar/ Palatal	Velar/ Glottal	'
Stops	Voiceless	(p)	T	K	‘	
	Voiced	B	D	G		
	Ejective	Ph	x	Q		
	Implosive	Dh				
Affricatives	Voiceless	Ch				
	Voiced	J				
	Ejective	c				
fricatives	Voiceless	F	s	Sh	h	
	Voiced	(v)	-	Nasals	m n ny	
Approximants		W	l	Y		

Flap/Trill	R

Table 2.2. Afaan Oromo Consonants (Dubbifamtoota)

2.2.3. Language Structure

Word Structure

The word, in Afaan Oromo "jecha" is the basic unit of a language. It is also a unit of language comprises one or more sounds that can stand independently and make sense. According to [9] the words of Afaan Oromo may run from very few monosyllabic words to polysyllabic words up to seven syllables. The writing system of the language is straightforward, that means, it is written as it is read and read as it is written. Afaan Oromo words are also differentiated from one another by white space characters. Hence, the task of taking an input sentence and inserting legitimate word boundaries, called word segmentation (tokenization) for information retrieval purpose, is performed by using the white space characters.

Sentence Structure

Afaan Oromo and English are different in their sentence structure. Afaan Oromo uses subject-object-verb (SOV) language because it has a flexible or variable word order Subject-verb-object (SVO) is a sentence structure where the subject comes first, the verb second and the third object. For instance, in the Afaan Oromo sentence "Tolaan barataa dha" or Preferably "Inni barataa dha" or "Baraataa dha Tolaan"?. "Toollan "is a subject, "barataa" is an object and "dha" is a verb. For that reason, it has SOV structure. The translation of the sentence in English is "Tola is a student" which has SVO structure because it has a fixed word order. There is also a difference in the formation of adjectives in Afaan Oromo and English. In Afaan Oromo adjectives follow a noun or pronoun; their normal position is close to the noun they modify while in English adjectives usually precede the noun, i.e. Afaan Oromo adjective agree with its head noun, this is not the case in English. For instance, miicaayyoo bareeduu (beautiful girl), bareeduu (adj.) follows miicaayyoo (noun).

2.2.4. Grammar (Gender, Number and Articles)

Every language has its own rules and syntax. Afaan Oromo also have its own grammar ("seer-luga" in Afaan Oromo).

Gender

Afaan Oromo has two genders, feminine and masculine in similar to other Afro-Asiatic language. The language use **-ssa** for masculine and **-ttii** for feminine because Some nouns and adjective are used us a noun to indicate a gender. For example: Ogeessa (expert) takes-ssa(male) and Ogeettii (expert) takes-ttii(female)[30].

Natural female gender corresponds to grammatical feminine gender as in the case of Aduu (sun), Urjii (star) and other astronomic bodies. In some Afaan Oromo dialects geographical terms such as names of towns, countries, rivers, etc are feminine, in other dialects such terms are treated as masculine nouns. It is due to this fact that there are different subject forms for the noun biyya 'country'. Example: biyyi(m.) or biitti (f.) .There are also suffixes like -a, -e that indicate present and past form of masculine markers respectively. -ti and -tii for present feminine marker and -te past tense marker, -du for making adjective form [30]. Biiftun baate 'the sun rose'. The word baate takes -te to show feminine gender. We can see that -tii can also show feminine gender in the following statement. Adurreen maal ariitii? What does the cat chase after? [32].

Number

In Afaan Oromo there are plural and singular numbers. It has different suffixes to form the plural of a noun. The use of different suffixes differs from dialects to dialects. In connection with numbers the plural suffix is very often considered unnecessary: miilla ishee lamaaniin with her two leg(s). According to Meiws[32] the majority of plural nous are formed by using the suffixes – (o)ota, followed by –lee, -wwan , -een, -olii/ -olee and –aan. Other suffixes like –iin in sariin dogs are found rarely. In addition, possessions, cases and article markers are often indicated through affixes in Afaan Oromo [23] [32].

-o(ota)	barsiiso(o)ta	teachers

-wwan	waggaawwan	years
-lee	gaaffilee	questions
-aan	ilmaan	sons
-olii/olee	Jaarsolii/jaarsolee	elders
-een	laggeen	rivers

Table 2.3. Afaan Oromo Gender (number)

Articles

In Afaan Oromo language there is no special word class of articles that appeared before a noun, like they exist in English. In English there are three main semantic choices for article insertion: definite article (the), indefinite article (a, an, some, any) and no article. In Afaan Oromo, however, the last vowel of the noun is dropped and suffixes (-icha,-ittii,-attii) are added to show definiteness instead of using definite article. For example, "haritti" is "the lake" to indicate certainty [32]. In some Afaan Oromo dialects the suffix **-icha** for male and **-ittii(n)** for female which usually has a singularize function is used where other languages would use a definite article[32].

Example: **Jaarsichi** the old man (Subject)

Jaartittiin the old women (Subject)

2.2.4. Conjunctions, prepositions, adjectives and Punctuation marks

Conjunctions

Conjunctions are used to connect words, phrases or clauses in a sentence. In Afaan Oromo there are different words that are used as a conjunction. According to Meiws[32] the main functions of conjunctions are indentified as: the function of coordinating clauses (coordination), the function of coordinating parts of sentence (coordination) and the function of coordinating syntactical unequal clauses(subordination) . Conjunctions in Afaan Oromo are "fi", "yookin", "haa ta'u malee", "garuu", "akkasumas" "kanaafuu". Example: Shaashamanne, Finfinnee fi Jimmaan magaalota Oromiyaati. (Shashamanne, Finfinnee and Jimma are Oromian cities.)

Prepositions

A preposition in Afaan Oromo connects nouns, pronouns and phrases to other words in a sentence. The word or phrase that the preposition introduces is called the object of the prepositions. Most Afaan Oromo prepositions are used in similar way it used in English. They are written separately from root word so, it is easy to remove from content bearing terms easily as a stop word. But, in some cases propositions may exist as linked with root words/stems. Prepositions in Afaan Oromo are "gara", "akka", "hamma", "booddee/booda", "dura", "ittaane", "malee", "keessa", "bira". For instance, ani gara shaashamanne deemee ture. (I went to shashamane.)

Adjectives

An adjective is a word which explains or modifies a noun or pronoun in the sentences. A modifier is a word that limits, changes, or alters the meaning of another word in the sentences. Unlike English, adjectives are usually placed after the noun in Afaan Oromo. Adjectives in Afaan Oromo are very important because its structure is used in every day conversation. For instance, in Sangaa adii bite "He bought white Ox" the adjective adii comes after the noun Sangaa [10].

Punctuation marks

Punctuation marks are placed in text to make meaning clear and reading easier. Afaan Oromo texts used different punctuation marks follow the same punctuation pattern used in English and other languages that follow Latin Writing System [31].It is used in both Afaan Oromo and English languages are the same and used for the same purpose with the exception of apostrophe. Apostrophe mark (') in English shows possession but in Afaan Oromo it is used in writing to represent a glottal (called hudhaa) sound. It plays an important role in the Afaan Oromo reading and writing system. For example, it is used to write the word in which most of the time two vowels are appeared together like "**ba'e**" to mean "get out", **"du'a"** to mean "death" and **"boba'aa"** to mean "fuel" with the exception of some words like **"bal'ina"** to mean "wide" which is identified from the sound created.

2.2.5 Synonymy and Polysemy

Synonymy refers to the fact that the same underlying concept can be described using different terms [13]. The phenomenon of synonymy is sufficiently widespread to account for the popularity. The notion of synonymy has a deceptively simple definition different lexeme with the same meaning.

The synonyms can substitute for one another in a sentence without changing either the meaning or the acceptability of the sentence in the document [13]. In LSI, the concept in query as well as all documents that are related to it is all likely to be represented by a similar weighted combination of indexing variables.

As an example: *chaaltuun mana qulqulleessuu jalati*. "Chaltu likes to clean house".

The underlined word *qulqulleessuu* can be substituted by the word "*Axawuu*" or *"haruu"* which cannot be used in this sentence. These three words can replace each other without changing the meaning of the sentence. In this way, three lexemes share a central core meaning.

Polysemy

Polysemy describes a single word that has more than one meaning, which is common property of language. Large numbers of polysemous words in the query can reduce the precision of a search significantly. By using a reduced representation in LSI, one hopes to remove some "noise" from the data, which could be explained as rare and less important usages of certain terms. The idea of polysemy allows to state that sense is related to, and probably derived from, without asserting that it is a distinct lexeme [13].

For example: *Qorachuun mala dhayiiti*. The study is the way of putting solution.

Here the word underlined *Qorachuun* which mean *Iyyaafachuu* "call out" and *gaafachuu* "ask" are two related meanings according the contexts of the sentence. As one suspect, the task of distinguishing homonymy from polysemy is not quite a straightforward. There are two criteria that are typically invoked to determine whether or not the meanings of two lexemes are related or not: the history, or etymology, of the lexemes and how the words are conceived of by native speakers [13]. In the absence of detailed etymological evidence, a useful intuition to use in distinguishing homonymy from polysemy is the notion of coincidence. On the other hand, it is far more difficult to accept cases of polysemy as coincidences.

2.3 Information Retrieval and Automatic Indexing

The need to store and retrieve written information became increasingly important over centuries, especially with inventions like scientific paper and books. Soon after computers were invented, people realized that they could be used for storing and automatically retrieving large amounts of information. In 1945 Vannevar Bush published a ground breaking article titled "As We May Think" that gave birth to the idea of automatic access to large amounts of stored knowledge. In the 1950s, this idea materialized into more concrete descriptions of how archives of text could be searched automatically [2][3]. Several works emerged in the mid 1950s that elaborated upon the basic idea of searching text with a computer. Several key developments in the field happened in the 1960s. Most notable was the works of Gerard Salton and his students; they were laid the foundation for retrieval system. They developed an evaluation technique for retrieval systems that is still in use by IR systems today [3].

The increase of electronic information inside and outside of the industry and available on-line would need following better techniques to access this information. Information retrieval (IR) is process of looking for material (usually documents) of an unstructured nature that satisfies an information need of the user from within large collections (usually stored on computers) [2]. Information retrieval can also cover other kinds of data and information beyond textual document; it could be image, multimedia or other data. The main concept in IR is finding relevant resources on the network that satisfies user information needs. Computerized system that can carry out information retrieval is known as information retrieval system [2].

2.3.1 Information Retrieval Process

There are two main processes in information retrieval; indexing and searching. The first process is before retrieval process can even be initiated, it is necessary to define document's representational terms set. These terms set comprise the logical view of the documents or abstraction of the whole document. With this term set index is constructed. Indexing is a critical data structure; it allows fast searching over large volume of data. Different indexing structures might be used but the most popular and common one is inverted file. Once the indexing constructed, it is amortized by querying the retrieval system proceeds [1].

The second process of information retrieval is retrieval process, the user first specify information need which then parsed and transformed by applying text operation on it[1][9]. The query is then

processed to obtain the retrieved documents. Fast query processing is made possible with the aid of index structure [1].

2.3.2 Automatic Indexing

Automatic indexing is the ability to representation of the documents or queries that could describe text documents based on their contents of the document by using computer. When the number of documents increases exponentially with the proliferation of the Internet, automatic indexing will become critical to maintaining the ability of the system to retrieve the information items sought by users of the system.

There are two major approaches for the automatic indexing of text documents: statistical approaches that rely on various word counting techniques, vector space model used this approach. The aim of statistical indexing is to capture content bearing words which have a good discriminating ability and a good characterizing ability for the content of a document. And those terms are going to be counted and will have some weight [7].

The other approach is linguistic approach that involves syntactical analysis [17]. It is based on knowledge base of the language it is working on [7]. Therefore, it needs ontology of the language to determine the terms which describe a document. It can use the extraction of the semantics the document using content bearing words extracted from document that going to be indexed and to resolve word ambiguities and/or generate relevant synonyms or polysemous based on the semantic relationships between words [7].

Semantic indexing

To enhance the performance of a traditional keyword-based search, a web documents should be represented with their concept rather than “bag-of-words”. But most of the earlier works on indexing and information retrieval depend on lexical analysis and statistical technique. Using these techniques, it is not easy to abstract the semantics of the documents [36]. Typically, information is retrieved by literally matching terms in documents and in query. Because there are usually many ways to express a given concept known as synonymy, the literal terms in a user's query may not match those of a relevant document. In addition, most words have multiple meanings known as polysemy, so terms in a user's query will literally match terms in irrelevant documents. A better approach would allow users to retrieve information on the basis of a conceptual topic or meaning of a document [13].

Latent Semantic Indexing (LSI) tries to overcome the problems of lexical matching by using statistically derived conceptual indices instead of individual words for retrieval. LSI assumes that there is some underlying or latent structure in word usage that is partially obscured by variability in word choice. A truncated singular value decomposition (SVD) is used to estimate the structure in word usage across documents [13].

The other method to solve lexical matching problem is Lexical chains technique for the extraction of concepts from the document and represents by concept vectors. And then text vectors, semantic indexes and their semantic importance degree are computed. This indexing method has an advantage in being independent of document length because we regarded over all text information as a value 1 and represented each index weight by the semantic information ratio of overall text information [36].

2.3.3 Semantic Index term extraction

Terms that are extracted from textual document are known to be a significant and basic linguistic descriptor of documents [4]. The Extracted terms are used for representing contents of the specific document. The terms extracted from the document can be used for indexing; it helps to distinguish a document from others documents [5]. In many cases, the terms that best describe the contents of a document are at the same time terminological units of the text‘s domain [5]. Using a computer to extract a key term from the textual document for indexing purpose is called automatic index text extraction. It is a critical constraint for many text related applications such as text clustering, indexing and others [4].

There are two main kinds of approaches to automatic index text extraction, statistical method and linguistic method [4]. Statistical method mostly relies on word frequencies: words that are repeated frequently within a document are likely to be good descriptors of its content. On the other hand, terms that occur in several documents do not have capability to distinguish one document from another [5]. It can be computed for a given term by multiplying its frequency in the current document (TF = term frequency) with its inverse document frequency (IDF) [5].

Linguistic method relies on syntactic criteria and do not use any morphological processes. Most researchers seem to agree that terms are mainly noun phrases to represent semantics of the text [4]. Mostly noun compounds, including adjectives verbs and sometimes with very small proportions [4].

2.3.4 Automatic Term Extraction and Weighting

Automatic indexing process can be considered to be composed of two tasks: first assigning terms or concepts capable of representing the content of each document in the database, called term extraction and second, assigning to each term a weight or values that signifies it importance for purpose of content description, called term weight[5][12]

2.3.4.1 Index Term Extraction

The task of extracting content descriptors itself is composed of a sequence of many activities like text preprocessing [5].

Lexical Analysis

Lexical analysis involves identification of individual words that constitute the input text. It is a process of collecting tokenized terms which needs some other operation on it and it may be used for indexing[2][3]. The tokens are taken from the document which may or may not be word.

Use of stop-List

Elimination of stop words is the process of avoiding a word which has fewer relevancies to determine the content of the document. Sometimes, some extremely common words which would appear to be of little value in helping select documents matching a user need are excluded from the vocabulary entirely. In other word it is non content bearing words. In English language, these words are most of the time articles, prepositions, sometime conjunctions and others. Therefore elimination of stop word is very critical task to control the size of the vocabulary [2][3]. Like other languages, Afaan Oromo has its own stop words. Usually words such as (example ‘as’, ‘achi’), conjunctions(“’fi’, ‘akkasumas’, ‘kana malees’, ‘Haa ta’u malee’) and are most frequent terms which are common to every document, and have no discriminating power one document from the other document. Therefore these terms should not be considered in indexing process [2].

Stemming

Stemming is the computational process of finding the root of the words or conflations. Conflation is occurrence of same word in little modification in its morpheme. The main idea behind stemming is that word with the same stem are semantically related and have the same meaning to the user of the text. Stemming is language dependent process in similar way to other natural language processing techniques. For example a word retrieve can be written as retrieve, retrieved, retrieving, retriever, and others. So this different form of the same word comes to its root term [2][3].

2.3.5 Term Weighting

All the terms included in the index list are not equally important in reflecting the content of a specific text. The main reason of computing term weight is to assigning weight depending on their level of importance [15]. Weighting index terms increase the precision of retrieval [1]. The functions use these distribution statistics to compute the weight of each term in each document and query [13]. The major weighting functions that are widely used in IR environment for frequency factors are discussed below.

I. Term frequency

Term frequency (tf) is the number of times a given term appears in that document. A document that mentions a query term more often has more to do with that query and therefore should receive a higher score. Since, assign to each term in a document a weight for that term that depends on the number of occurrences of the term in the document [2]. This concept can be applied through assigning the weight; it is equal to the number of occurrences of term t in document d. This weighting scheme is referred to as term frequency and is denoted $tf_{t,d}$, with the subscripts denoting the term and the document in its order [2]. The exact ordering of the terms in a document is ignored but the number of occurrences of each term is material. We only tried to capture information concerning to the number of occurrences of each term. [2].

$$Tfij = \frac{fij}{\max\{fij\}} \quad \text{.....2.1}$$

II. Inverse document frequency

It will create a problem when we make all terms are equally important; the impact would be explicitly viewed when it comes to assessing relevancy against a query [2]. In fact certain terms have little or no discriminating power in determining relevance. For instance, a collection of documents on the auto industry is likely to have the term auto in almost every document. To this end, we introduce a mechanism for attenuating the effect of terms that occur too often in the collection to be meaningful for relevance determination. An immediate idea is to scale down the term weights of terms with high collection frequency, defined to be the total number of occurrences of a term in the collection. The idea would be to reduce the tf weight of a term by factor that grows with its collection frequency [2]. The main objective to IDF values, it can increases importance

proportionally to number of times a word appears in the document but is offset by the frequency of the word in the corpus.

document frequency of the term, ***dft***, defined to be the number of documents in the collection that contain a term t. its aim is to obtain document level statistics, which deals with the number of documents containing a term [2]. Using ***dft*** is difficult to measure the discrimination power of the term among the documents. To come over this problem, it better to use inverse document frequency, which is logarithmic function of total number of the document N and over dft[2]. It is defines as

$$\text{idf} = \log_2 N/dft \quad \ldots\ldots\ldots\ldots\ldots\ldots 2.2$$

III. Tf-idf weighting

This weight is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. So we can figure out the importance of the keywords in the documents and then we can find out which keyword occurrence. Combining the definitions of term frequency and inverse document frequency helps to produce a composite weight for each term in each document. The ***tfidf*** weighting scheme assigns to term ***t*** in document ***d*** given by

$$\text{Tf-idf}_{t,d} = \text{tf}_{t,d} * \text{idf}_t \quad \ldots\ldots\ldots\ldots\ldots\ldots 2.3$$

Tf-idf t,d assigns to term ***t*** is a weight in document ***d*** that is

i. Highest when t occurs many times within a small number of documents (thus lending high discriminating power to those documents);

ii. Lower when the term occurs fewer times in a document, or occurs in many documents (thus offering a less pronounced relevance signal);

iii. lowest when the term occurs in virtually all documents.

2.3.6 Query matching model

Comparison of query and document representations is very important activity of IR system; to achieve in order to retrieve documents that are more similar to the specific query. The three classic models of Information retrieval: the Boolean model, the Vector space model and the probabilistic

model are often used to accomplish this particular task [1]. The Boolean, vector space model and probabilistic models are briefly discussed below.

The Boolean Model

Boolean model is the oldest of the three classic retrieval models and it relies on the use of Boolean operators in combination with set theory [1]. The Boolean model considers that index terms are present or absent in a document. As a result, the index term weights are assumed to be all binary i.e {0,1}. A query is usually a Boolean expression of words input directly by the user. The query can be converted into a Boolean expression taking from a user's query free sentence. The terms in a query are linked together with AND, OR and NOT [12]. This method is often used in search engines on the Internet, because it is fast and can therefore be used online [12]. In Boolean model there is no relevance judgment and ranking mechanism because it predicts that each document is relevant or non- relevant.

Unfortunately, the Boolean model has got its own drawbacks. It requires the users to have some knowledge of the search topic for the search to be effective [1]. A wrong word in a query could rank a relevant document non-relevant. In addition to that, all retrieved documents are considered to be equally important.

Another drawback of the Boolean model is that, most users find it difficult to translate their information need into a Boolean expression [1]. This means most users need an intermediary to work with the Boolean retrieval model, which in turn brings its own problems, such as misunderstanding and possibly misrepresentation of the information needs of the user.

The Vector space model

In vector space model document and query, are represented as a vector of weights. A vector space is determined by all the index words selected from the entire document collection [12][7]. A value in a document (query) vector describes the degree importance of the corresponding word (term) in representing the semantics of in that document (query). Sometimes a given term may receive a different weight in each document in which it occurs. If the term is not appearing in a given document the weight of that term will be zero in that document .In other words, given a vector space as follows [12][1]:

Vector space: < ***t1, t2, ..., tn*** >

A document and a query may be represented in the following vectors of weights:

d → <***wd1, wd2, ..., wdn***>

q →<***wq1, wq2, ..., wqn***>

Where ***wdi*** and ***wqi*** are the weights of ***ti*** in document ***d*** and query ***q***. Query matching involves measuring the degree of similarity sim(***d, q***) between the query vector q and each document vector ***d***. Equation 2.4 is a formula of cosine similarity[12][8]:

$$sim(d_i, q) = \frac{\vec{di} \cdot \vec{q}}{\left|\vec{di}\right|\left|\vec{q}\right|} = \frac{\sum_{i=1}^{n} w_{di} * w_{qi}}{\sqrt{\sum_{i=1}^{n} w_{di}^2} * \sqrt{\sum_{i=1}^{n} w_{qi}^2}} \text{----------2.4}$$

Again, the documents with the highest degrees of similarity are the answers to the query.

The Probabilistic Model

In the probabilistic model, the retrieval is based on the Probability Ranking Principle. The probability ranking principle asserts that the best retrieval effectiveness will be achieved when documents are ranked in decreasing order according to their probability of relevance [2]. In other words it creates an ideal answer set. This means that the querying process is a process of specifying the properties of the ideal answer set which is not known exactly.

Given a user query, the document collection can be divided in to two sets; a set which contain exactly the relevant documents and a set which contains the non relevant documents [2]. Therefore, the querying process can be seen as specifying the properties of the relevant document set using index terms. Though, since the properties of the relevant document set is not known in advance, there is always the need to guess at the beginning the descriptions of this set. The user then takes a look at the retrieved documents for the first query and decides which ones are relevant and which ones are not. By repeating this process iteratively the probabilistic description of the relevant document set can be improved [2].

The feature of the probabilistic model to rank documents in decreasing order of their probability of relevance to the user query is taken as the most important feature of the model[2][37]. However, lack of initial information on the relevant and non-relevant documents in the collection with respect to specific query; its ignorance to the frequency with which an index term occurs in a document and the assumption of index terms independence are taken as its major drawbacks[2][37].

2.3.7 IR System Evaluations

One of the usual and challenging tasks to be carried out in any research is the evaluation of the system. In this work, Recall and Precision are two commonly used metrics to measure the retrieval performance of IR systems [1][13]. Recall is defined as the proportion of relevant documents that the retrieval system found and precision is the proportion of documents that the system found that are relevant.

To show these metrics, suppose we have a document collection D, a query Q, and a retrieval system S. Out of the documents in the collection, let us assume that R of them is relevant to query Q. The relevance of a document to a query is often determined by domain experts. Finally, let us suppose that system S retrieves a set of 'n' documents for Q and 'r' of the 'n' retrieved documents is relevant to Q. The recall and precision for Q and D are defined as:

$$Recall(S) = \frac{r}{R} \quad \text{.....2.5}$$

$$Precision(S) = \frac{r}{n} \quad \text{.....2.6}$$

Where **'r'** is number of relevant documents retrieved, **'R'** total number of relevant documents in the database and **'n'** is total number of documents retrieved.

On the other hand, most modern IR systems cannot return a set of documents for a query, hence, directly apply these formulas is not possible. Instead, they output a list of documents ranked in descending order of probability of relevance to the query [1][13]. To come up with a solution to this problem, recall/precision pair is calculated for each relevant document in the ranked output and then interpolate to find the precision at standard recall points based on the given information that recall-precession curve is constructed. Hence, precision for recall point 0.0 means the precision for the first relevant document in the ranked output. Often the average of the precision values at the standard recall points is used as a single metric to measure the quality of the ranked output [1][14]. When a set of query is considered, the precision at a recall point for the query set takes the average of the

precision values of all queries at that recall point. The average precision for the query set takes the average of the average precision values of all queries.

2.4 Latent Semantic Indexing (LSI) IR

Typically, the classic information retrieval models (Boolean, standard vector space and probabilistic) are used to finding relevant textual material depends on matching individual words in users' requests with individual words in database texts. Usually text document that contain one or more words in common with those in the users' query are returned as relevant. Keyword or content-based retrieval systems like this are, however, many document relevant to a users' query are missed, and many irrelevant or unrelated materials are retrieved [12], [14]. So, individual term matching methods can be inaccurate when they are used to match a user's query. However, there are a number of issues which can prevent the results of an exact term matching search from being accurate. Two of these issues include synonymy (people to use different ways to refer to the same thing) .For instance, in Afaan Oromo the word **'Yaaduu', 'Herreguu'** and **'xiinxaluu',** all can be used to describe **'Think'** (from Afaan Oromo dictionary entitled, "Galmee jechoota Afaan Oromoo".). Thus, in a literal search, only documents containing the literal query terms will be matched, causing the search results to exclude relevant documents [12]. Therefore, synonyms tend to reduce the recall of IR systems. And polysemy (words are characterized by different meaning based on the context). For instance, the word **"daakuu"** in Afaan Oromo can mean **'flour'**, or it can also mean **'swimming'**. So terms in a user's query will literally match terms in irrelevant documents [12]. Thus, cause terms in the query to be matched with words in irrelevant documents and for the search results to be too broad and underlying poor precision [12].

Latent Semantic Indexing (LSI) [12] tries to overcome the problems of lexical matching by using statistically derived conceptual indices instead of individual words for retrieval. LSI assumes that there is some underlying or latent structure in word usage that is partially obscured by variability in word choice. A truncated singular value decomposition (SVD) is used to estimate the structure in word usage across documents. Retrieval is then performed using the database of singular values and vectors obtained from the truncated SVD. Performance data shows that these statistically derived vectors are more robust indicators of meaning than individual terms.

Dimensionality Reduction: - Dimensionality reduction technique seeks to resolve the problems of synonymy and polysemy by examining the "latent" structure of a document and the terms within it

[12] [13]. These techniques decompose words and documents into vectors in a low dimensional space.

The variability in word choice between the system users and authors of documents is addressed, because any word can be matched with another word to some degree in the low dimensional space [12] [13].

2.4.1 Latent Semantic Indexing Approach

Latent semantic indexing is one of the many recent information retrieval techniques that are based on a vector space model [13] [14]. However, LSI adds additional processing on the term by document matrix, maybe the weighted one, generated by standard vector space model. The term by document matrix is decomposed into a set of 'r', rank of the matrix, orthogonal factors from which the original matrix can be approximated by linear combination. Typically, any rectangular matrix, X, for instance, a tXd matrix of terms and documents can be decomposed into the product of three other matrices [34].

$$X = U \Sigma V^T \text{ or}$$

$$X = Uo * So * Vo^T \text{---------- (2.7)}$$

$$tXd \quad tXt \quad tXr \quad rXd$$

such that columns of Uo[u1,u2,u3,…u_r] is a tXr matrix that define the left singular vectors of matrix X and Vo[v1,v2,v3,…v_r] is a sXr matrix that define the right singular vectors of matrix X, which are the orthogonal eigenvectors associated with XX^T and X^TX respectively. So is a diagonal matrix representing the singular values of X arranged in decreasing order, which are non-negative square roots of the Eigen values of X^TX [12][14]. This is called Singular Value Decomposition (SVD) of X. If only the 'k' largest singular values of So (where K<= r (rank of A)) are kept along with their corresponding columns in the matrices Uo (u1, u2… uk) and Vo(v1, v2 … vk), and the rest avoided, yielding matrices Uk, Sk and Vk , the resulting matrix Xk is the unique matrix of rank 'k' and is closest in the least squares sense to the original matrix X [12] [14][34].

$$X \sim Xk = Uk * Sk * Vk^T \text{---------- (2.8)}$$

tXd tXd tXk kXk kXd

The point here is that the reduced matrix Xk, by keeping only the first 'k' independent linear components of X, captures the major associational structure in the matrix and also much of the noise caused by, such as, polysemy and synonymy, that leads to poor information retrieval is removed with the reduction in dimensionality [13].

Hence, the selection of the right dimensionality or the value of 'k' is a crucial issue. Of course, the value of 'k' should be large enough to integrate all the real structure in the document collection, but also it should be small enough so that noises that are caused by the variability in word usage are not included [12] [13].

There is no hard and fast rule that can be used determine the optimal value of 'k'. Currently, an operational criteria is used, that is, a value of 'k' which yields good retrieval performance [13]

In this reduced model, the similarity of documents is determined by the overall pattern of term usage instead of exact term match, so documents can be near each other regardless of the precise words that are used to describe them [12]. A document which has no words in common with a user's query may be near to the query if that is consistent with the major pattern of word usage.

Geometrically, the location of terms and documents in the approximated k dimensional space is given by the row vectors from the Uk and Vk metrics respectively [13]. The cosine or dot product between vectors in this space corresponds to their estimated similarity. The goal is not to describe the concepts verbally, but to be able to represent the documents and terms in a unified way for exposing document-document, document-term, and term-term similarities. The representation of both term and document vectors in the same space makes the computation of the similarity between any combination of terms and documents very easy. From IR point of view, three basic similarity comparisons in the reduced matrix Xk are important [12] [13]

I. Term by Term Comparison

The term or rows of the vectors, comparison is used to determine the extent to which two terms(term to term) have a similar pattern of occurrence across the document. Using the reduced matrix Xk in equ (2-6) and the dot-product similarity measure, the term-to-term similarity values can be computed as

$$Xk \cdot Xk^T = (Uk.Sk.Vk^T).\,(Uk.Sk.Vk^T)^T = Uk \cdot Sk \cdot Vk^T \cdot Vk \cdot Sk \cdot Uk^T$$

$$= Uk \cdot Sk \cdot (Vk^T \cdot Vk) \cdot Sk \cdot Uk^T$$

$$= (Uk.\ Sk).(Uk.Sk)^T \text{---------- (2.9)}$$

Here, we can see that the i, j cell of the square matrix $Xk \cdot Xk^T$ can be obtained by taking the dot-product between the ith and jth rows of the matrix Uk . Sk. We can consider Uk. Sk as coordinates for terms, because since Sk is a diagonal matrix, it is merely used to stretch or shrink the axis of the reduced vector space without affecting the position of Uk in the space.

II. Document By Document Comparison

The analysis is used to determine the similarity between the documents by identifying the extent to which the two documents have a similar term occurrence pattern [12]. This can be achieved by computing the dot-product of column vectors of the reduced matrix Xk i.e.

$$Xk^T \cdot Xk = (Uk \cdot Sk \cdot Vk^T)^T \cdot (Uk \cdot Sk \cdot Vk^T)$$

$$= Vk \cdot Sk \cdot (Uk^T \cdot Uk) \cdot Sk \cdot Vk^T$$

$$= (Vk.Sk)(Vk.Sk)^T \text{---------- (2.10)}$$

The i, j cell of $Xk^T \cdot Xk$, which is a document-by-document square matrix, is obtained by taking the dot product between the i^{th} and j^{th} row of the matrix as coordinates for documents, Vk.Sk. Though, the space of Vk . Sk is just a stretched or shirked version of the space of Vk, in proportion to the corresponding diagonal elements of Sk.

III. Term by Document Comparison

The third comparison is between a term and a document. However, the comparison between a term and a document is just the value of an individual cell of Xk. Hence,

$$Xk = Uk.Sk * Vk^T \text{---------- (2.11)}$$

$$= Uk \cdot Sk^{½} \cdot Sk^{½} Vk^T = (Uk \cdot Sk^{½})(Vk \cdot Sk^{½})^T$$

This implies that the i, j cell of Xk is obtained by taking the dot-product between the i^{th} row of the matrix Uk .Sk½ and the j^{th} row of the matrix Vk .Sk½ (or the j^{th} column of $(Vk \cdot Sk^{½})^T$). Here again, the effect of the factor Sk½ is stretching or shrinking of the axes by a factor of its value.

2.4.2 Query Representation

One of the major challenges In Information retrieval is the description and representation of information needs in terms of a query. Among the central and essential feature of information

retrieval is matching the text of the query to the text of the document in the corpus. In LSI based IR systems, a user's query is often considered as a pseudo document and is represented as a vector in the reduced term-by-document space [12] [13]. First, the terms used by the searcher are represented by an (m X 1) vector 'q' whose elements are either zero or correspond to the frequency of terms that exists in the database of reduced vector space. The local and global term weights used for the document collection are applied to each non-zero elements of the query vector q. Then the query vector is represented in the reduced LSI space by the vector

$$\hat{q} = qT.Uk.Sk^{T} \text{---------- (2.12)}$$

where q^{T}**.** Uk is the sum of term vectors precise by vector q scaled by Sk^{T} [12] [13]

Finally the query vector can then be compared to all existing document vectors, and the documents ranked by their similarity (nearness) to the query. One common measure of similarity is the cosine between the query vector and document vector. Typically, the first 'n' closest documents or all documents exceeding some cosine threshold are returned to the user [12].

2.5 Related Research works

So far, the area of information retrieval has grown well beyond its primary goals of indexing text and searching for useful documents in a collection [1]. In order to increase the efficiency and effectiveness of retrieval system, there are different techniques with different models have been used. Seeing as meaning or concepts behind the words are an important issue in Information Retrieval area, that why most scholar to deal how to solve the problem. In IR research, Latent Semantic Indexing has received significant attention and previous work was done to discover methodologies that can improve the performance of information retrieval system using different latent semantic techniques.

2.5.1 IR Systems for International Languages

Now a day's information retrieval system is highly connected with human every day activities. There are many search engines for searching text documents, audio, video, pictures, and software. In addition to that search engines like Amazon, which particularly works for online marketing of products and services. In western world several works are done in the area of information retrieval.

But the majority of the works have been done so far is for major technology languages in the world like, English and French.

Chinese is one of the largest language of the world by it is native speakers [34]. But the growth of Chinese language is far below of other language like English. The journal article entitled "Important Issues on Chinese Information Retrieval" that implies the importance of giving emphasis for Chinese information retrieval system especially in this is age of the Internet. The study is identified major problems that should be addressed including need of standard test collection, availability of Internet searching tools, key word indexing, document classification, and information filtering [35].

According to [36] Arabic alphabets are used for the writing system of an Arabic language. But it was un-common accessing documents written in Arabic language on the web until UNICODE standardization come in to existence. Since information retrieval is language dependent process Arabic has also its own unique characteristics. The problem with retrieving Arabic document is that search engines only retrieves just small amount of relevant documents available in search corpus. The other problem is non-Latin languages are not supported well by search engines. Moreover, available techniques are hard to implement for Arabic system. But it is agreeable on the importance of having IR system even when there are some challenges.

2.5.2 IR Systems for Local Languages

For local languages in Ethiopia the review has been done to find out the work done. But the work found by the reviewers involves only Amharic and Afaan Oromo related studies in the area of IR.

2.5.2.1 Amharic Retrieval Systems

Tewodros H[24] used latent semantic indexing strategy with singular value decomposition to develop Amharic text retrieval systems. Amharic is morphologically rich language that has lexical variation. His hypothesis is that vector space model decreases the effectiveness of the system in comparison to LSI. That is why the author preferred using LSI. The main advantage of using LSI is that it handles problems related with polysemy and synonymy. Non-content bearing terms and also stop-words were removed. Additionally term weighting technique has been used for measure importance of terms in the document. The term weighting technique used was tf*idf. Cosine similarity was used to measure similarity and dissimilarity. Better result obtained by using LSI. Achieved performance by using VSM approach is 0.6913 with 206 news articles and 9256 index

terms and by LSI approach 0.7157. This implies a 2.4% improvement over the standard vector space method.

The future direction stated by the author includes: if the size of the document increased, it would have a chance to improve the performance of the IR system. Furthermore the index terms used in the system were not stemmed, if they could be stemmed it will have possibility of enhancing the IR system performance.

Abey B[25] tried to articulate the problem that was clearly seen in the Amharic IR system, specially the impact of polysemy and synonymy of terms in the documents. Amharic has a rich verb morphology which is based on triconsonantal roots with vowel variants describing modifications to, or additional detail and variants of the root form .As a result of that different user may describe their information needs in different way, for that matter their queries may differ lexically. For that reason to overcome this problem the researcher proposed semantic based query expansion for Amharic IR system.

Abey B[25] has used text preprocessing to make avail the term in the document to indexed in well-organized manner. He was developing Vector space model to use for IR system. The expansion of the query given to the system based on the knowledge base of Amharic language or thesaurus. In his study the performance was evaluated using the usual IR system criteria, recall and precision. The performance of the system measured compared with standard vector space model. He was develop Vector space model to used for IR system

The performance of the system registered 0.92(92%) recall and 0.68(68%) precision. These shown us query expansion in Amharic Information Retrieval system registered better recall but the precision gone down. One of the future directions recommended by the author is to design ontology based query expansion in order to control expanding terms that are polysemous by themselves.

Mulualem W[26] developed semantic indexing and document clustering for Amharic information retrieval. His work mainly focuses on indexing and clustering process, to solve the problem found in semantic natures of the language. The system retrieves the information needs of the user by retrieving documents which have similar meaning to the query formulated by the user rather than exact terms match. Moreover, there was no any attempt to make retrieval effective through document clustering. In order to solve these issues, integrating of semantic indexing of documents

and document clustering techniques with generic IR system will improve the retrieval performance. It comprises three basic components indexing, clustering and searching. The system comprises all processes exist in generic IR plus to that C-value technique multi word term extraction, k-means algorithms document clustering, cluster base searching strategy used.

In his study the performance was evaluated using the most common and basic statistical IR system measurement criteria, precision, recall and F-measure. The performance of the system registered an average of 0.88(88%) recall, 0.51(51%) precision and 0.72(72%) f-measures for frequent queries string. And average 0.60(60%) recall, 0.42(42%) precision and0.60 (60%) f-measures for less frequent string queries. When we calculate the average performance over frequent and less frequent query strings, we found an average of 0.74(74%) recall, 0.46(46%) precision, and 0.66(66%) f-measures. These shown us semantic indexing and document clustering based Amharic IR system registered 6% of F-measure improvement. This means the performance of the Amharic IR system increases from F-measure of 60% (Amanuel) to F-measure of 66% accuracy.

The researcher believes that the performance of the system can be increased if context aware IR system using a well developed thesaurus to aware the context to which the terms in the text appears. In addition using well tagged corpus of bigger size could help to discriminate the context in which terms are presented.

2.5.2.2 Afaan Oromo Cross Lingual information retrieval System

Daniel B[27] used corpus based approaches to develop Afaan Oromo–English Cross Lingual Information retrieval systems. His work mainly focuses on to enable Afaan Oromo users to specify their information need in their native language and to retrieve documents in English. His work is based on a parallel corpus collected from Bible chapters, legal and some available religious documents for training and testing purpose. He used word based query translation strategy for the two language pairs, English & Afaan Oromo, because document translation is computationally expensive. The system is developed using a statistical word alignment tool called GIZA ++.

Daniel B[27]developed Afaan Oromo-English bilingual dictionary using 530 parallel documents and all the collected documents were used for the construction. In the collected data different pre-processing tasks is used for the word alignment tool and information retrieval task. The two mandatory input files are Vocabulary and bitext files for GIZA++ tool for the formation of the word

alignment. These files were generated by the packages available in GIZA++ toolkit. Then, as input for the GIZA++ to create word alignment statistical information of vocabulary and bitext file generated was used. In this way the author developed the bilingual dictionary and this dictionary served as translation knowledge source.

Experimentation was conducted in two phases. In the first phase the un-normalized edit distance was used to relate variation of words between query and index terms while for the second phase of the experimentation by using normalized edit distance. Evaluation of the system is conducted for both monolingual and bilingual retrievals. In the monolingual run, Afaan Oromo queries are given to the system to retrieve Afaan Oromo documents while in the bilingual run the Afaan Oromo queries are given to the system after being translated into English to retrieve English documents. The system evaluated by the two commonly known parameters, recall and precision. In order to evaluate the performance he was conducted for 60 queries and 55 randomly selected test documents. In the first phase of the experimentation, we are obtained 0.421 maximum average precision for Afaan Oromo documents and 0.304 maximum average precision for English documents while In the second phase of the experimentation, we are obtained 0.468 maximum average precision for Afaan Oromo documents and 0.316 maximum average precision for English documents. Therefore the second phase of experimentation performs slightly better than the first. From the experiment results, the researcher conducted that with the use of large and cleaned parallel Afaan Oromo-English document collections, it is possible to develop CLIR for the language pairs.

Eyob N[28] used corpus based approaches to develop Afaan Oromo–Amharic Cross Lingual Information retrieval systems. His work mainly focuses on to enable Afaan Oromo speakers to retrieve Amharic information using Afaan Oromo queries. His work is based on a parallel corpus collected from; news articles, Bible, legal documents and proclamations from customs authority were used for training and testing purpose. The translation strategy used in this work is word based and a python script was developed to accept Afaan Oromo queries and translate them into their Amharic equivalent by looking through the bilingual dictionary word by word. The system is developed using GIZA ++ word alignment tools.

Eyob N[28] developed Afaan Oromo-Amharic bilingual dictionary using parallel documents and all the collected documents were used for the construction. In the collected data different pre-processing tasks, like tokenization, normalization, stemming, is used for the word alignment tool and

information retrieval task. The most mandatory and minimum requirement input files for word alignment are the vocabulary file and the bitext file. These files were generated by the packages available in GIZA++ toolkit. Then, as input for the GIZA++ to create word alignment statistical information of vocabulary and bitext file generated was used. In this way the researcher developed the bilingual dictionary and this dictionary served as translation knowledge source.

The system is tested with 50 queries and 50 randomly selected documents. Two experiments were conducted, the first experiments is done by using only one possible translation to each Afaan Oromo query term and the second experiments is done by using all possible translations. The retrieval effectiveness of the system is evaluated by the two commonly known parameters, recall and precision for both monolingual and bilingual runs. Accordingly, the result of the first experiment showed a maximum average recall value of 0.58 and maximum average precision of 0.81 for the monolingual run; and maximum average recall of 0.38 and maximum average precision of 0.45 for the bilingual run. The result after conducting the second experiment returned a maximum average recall of 0.7 and a maximum average precision value of 0.6. The result of the second experiment showed better result of recall and precision than the first experiment. The result obtained in the second experiment is a maximum average precision of 0.60 for the bilingual run and the result for the monolingual run remained the same. From the second experiment, it can be concluded that using all possible translation can be used to improve the overall retrieval effectiveness of the system.

2.5.2.3 Afaan Oromo Text retrieval System

Gezahegn G[20]conducted a research which entitled Afaan Oromo Text Retrieval System. He aimed to come up with an Information Retrieval system that can enable to search for relevant Afaan Oromo text corpus. For the study 100 different textual documents were collected from different news media. The collected corpora involve different subjects like politics, education, culture, religion, history, social, health, economy and other events.

The designed system has two main components: indexing and searching. Once the corpus has been collected different pre-processing (tokenization, normalization and stemming) activities were employed on the documents to make them suitable for indexing. In the normalization process all the characters are converted to lower case and all the punctuation marks except the “ ‘ ” which have different meaning in Afaan Oromo, were removed by using python script. The stemming part of the

pre-processing is done by using a rule based stemmer developed by Debela T and Ermias A[35] which was based on the porter stemmer algorithm. The index file structure used in the study is inverted index. Inverted file index has two files Vocabulary file and Post file which were used in building vectors of document versus terms. Index terms should be content bearing words and for this task stop word list has been prepared manually. The term weighting technique used in the study is term frequency – inverse document frequency (tf-idf). The similarity measure is done by using the popular cosine similarity measure. The searching component is based on the Vector Space Model and this was implemented using python script.

For testing the designed system all the collected documents and 9 queries were prepared; and these queries are marked across each document as either relevant or irrelevant to make relevance evaluation. The study used precision and recall as measure of effectiveness. Results from the experiment returned an average performance of 0.575(57.5%) precision and 0.6264(62.64%) recall. The researcher believes that the performance of the system can be increased if stemming algorithm is improved, standard test corpus is used, and thesaurus is used to handle polysemy and synonymy words in the language.

Having seen some of the related works done so far mainly focusing on local languages, this paragraph will introduce this research work. So, this research focuses on Semantic-based Indexing for Afaan Oromo documents to improve the performance of Afaan Oromo IR system based on the recommendation of previous researchers and research works in other languages.

Chapter three

System design

3.1. Introduction

This chapter discusses the design and development of Semantic-based Indexing Information Retrieval for Afaan Oromo. The peculiar characteristics of Afaan Oromo language and writing system in relation to text retrieval as well as basic concepts in the SVD.

Various researchers had conducted researches to tackle problems exhibited in Afaan Oromo text retrieval and to enhance the performance of the retrieval system as discussed in section 2.5.2. The researches tried to overcome the problems of Afaan Oromo Text Retrieval systems using Vector Space Model Gezahegn G[20]. Though these and others research were conducted, still there are several problems with regard to Afaan Oromo IR. Because problem related with extract semantics of the documents. Plus to that inefficient way of search engine strategy makes the system to have less performance [2]. In this research the problems stated above are handled and tried to solve.

3.2. System Architecture

The development of an IR system involves various techniques and methods. The design of the prototype of Applications of Information Retrieval for Afaan Oromo text based on Semantic-based Indexing has the components depicted in figure 3.1 below. The different components represent the major activities involved in the prototype of Applications of Information Retrieval for Afaan Oromo text based on Semantic-based Indexing.

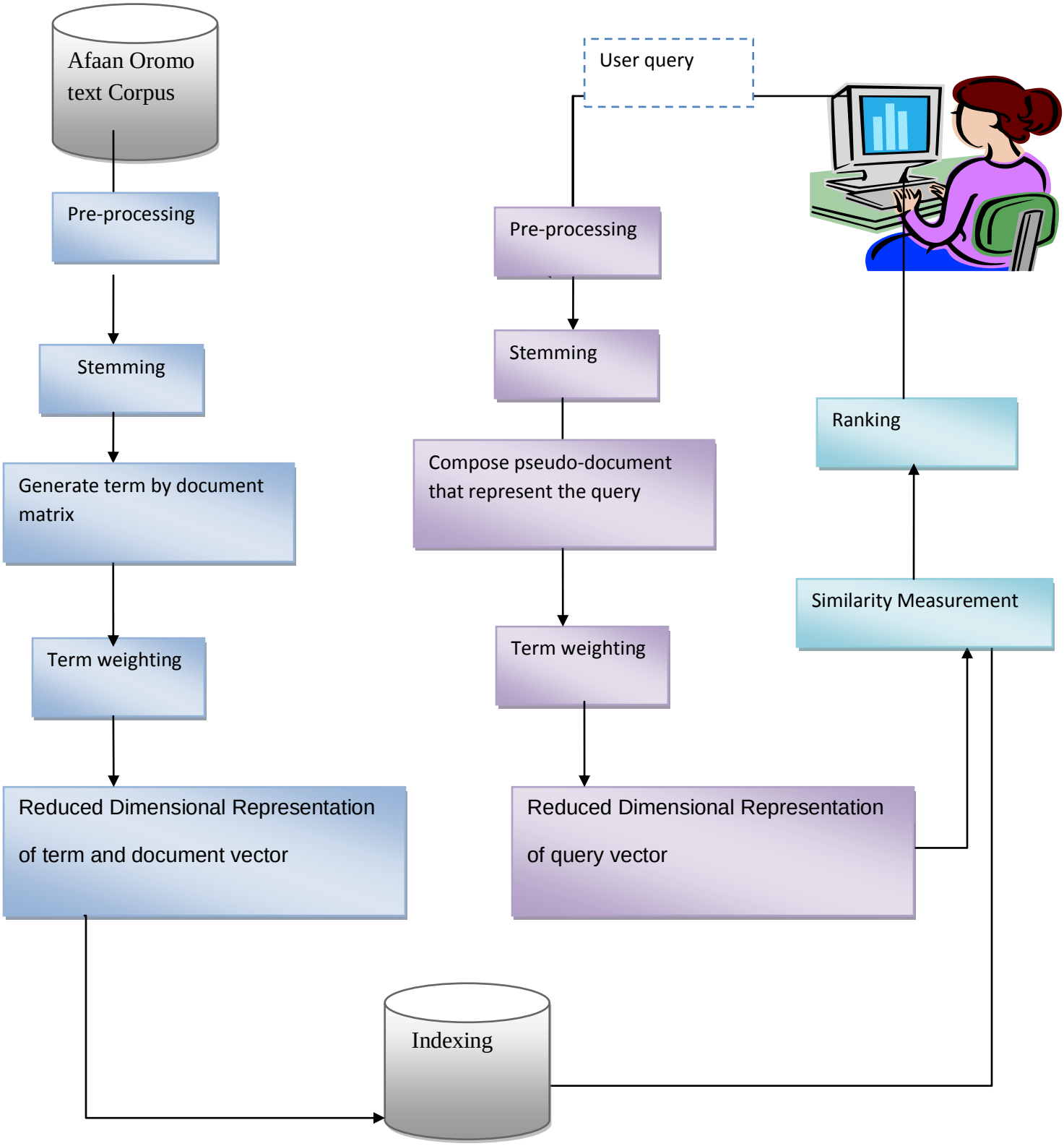


Figure 3.1 Architecture of the System

The main objective of this prototype model is to represent the terms, documents and queries in the similar reduced-dimensional space so that comparison of the documents, terms and queries in this reduced-dimensional space is possible. This makes the LSI model is straightforward.

For the experiment the collection of the document and the sample query files are stored in their respective folders are read and individual words which are not in the stop-word list are extracted. The term by document matrix and the query vector are generated. In order to increase or decrease the importance of each term based on their ability to represent the content of a document and also discriminate it from other documents in the collection; both the term-document matrix and the query vectors are weighted by using **tfidf** weights.

The weighted term-document matrix is then given to the SVD algorithm as an input and a reduced dimensional representation of the matrix is generated from it. The reduced-dimensional representation of the query vectors is also obtained and reduced into the reduced space. Because the concept space is ideally much lesser than the word space, we can limit the number of singular values such that important semantic information is not lost and noise is removed from the system. After the terms, documents and queries are represented in the same reduced dimensional space, the cosine similarity measure is used for identify the ranked list of relevant documents for each query of the system.

3.3 Data Collection and Acquisition

As there are no standardized corpuses in the area of information retrieval on Afaan Oromo, we have compiled our own data collected from different sources. The sources include newspapers, Bible, news (Oromia Radio and Television Organization (ORTO)), magazines, historical documents and Bulletins. The reason to widen the coverage of the source is to make it relatively balanced, representative of a language variety and avoid bias.

3.3.1 Corpus Preparation

This section deals with the collection of document and ways of preprocessing in the collected documents. For this thesis, the researcher prepared new corpus for the Applications of Information Retrieval for Afaan Oromo text based on Semantic-based Indexing. The size of the corpus is an important factor in the performance of the IR system; with the general point that the amount of data is big it has a better quality [2]. However, in such a small work, it is very difficult to obtain standardized data set for under resourced language like Afaan Oromo that why limited size of corpus

and limited number of test queries are used. In addition, this corpus needs further development in the other works. The data set is collected from newspapers (Bariisaa, Kallacha Oromiya and Oromiya. Bariisaa is a weekly newspaper, whereas the rest two come out once in two weeks, linguistics, ORTO news, government and official websites.

Moreover, Oromia Radio and Television Organization found in Adama releases daily news through radio and television broadcast and on its official website [39]. To reduce the data sparsity the researcher used data from these sources since they are believed to represent texts addressing various issues of the language.

Several preprocessing computation were applied on the collected documents to make them readily available to be being indexed and searched for the user need. After the document identified and collected the data preprocessing tasks are followed. The major preprocessing tasks are discussed below.

A. Tokenization

Tokenization refers to splitting the character streams into a set of tokens (usually terms). While linguistic preprocessing then deals with building equivalence classes of tokens which are the set of terms that are indexed. Tokenization in this work also used for splitting document in to tokens and detaching certain characters such as punctuation marks or non content bearing words[1]. This process detects the boundaries of written terms. Tokenizing of a given terms depends on the characteristics of the language of the terms in which it is written.

B. Stop word Removal

Not all terms found in the document are equally important to represent documents they exist in. Some terms are common in most documents. Therefore, removing those terms, which are not used to identify some portions of the document collection, is important. Like other languages, Afaan Oromo has its own stop words. Usually words such as (example ‘as’, ‘achi’), conjunctions(“’fi’, ‘akkasumas’, ‘kana malees’). Since stop words do not have significant discriminating powers in the document collection; the researcher filtered stop-words list to ensure only content bearing terms are included. Mostly, non-content bearing terms are removed by either removing high frequency terms from the indexing term list or by using a negative dictionary (stop word list) for that language[1][2].

Stop Words	Meaning (English)
Kana	This
Achi	There
Yookan	Or
Haa ta'u malee	However
Irra	On
Jala	Under
As	Here
Kanaaf	So
Jedhama	Called
Ammo	However, but
Garuu	But
Bira	Beside, at, near of
Ala	Outside, out

Table 3.1. Stop words of Afaan Oromoo

C. Stemming

In this work, the researcher has used stemming for many word variants/affixes. For computational efficiency and better retrieval performance it is crucial to conflate morphological variations. Thus, stemming of the Afaan Oromo corpus and query is done by using a rule based stemmer developed by Debela T and Ermias A [35] which has six rules. Stemming techniques are language dependent. Therefore, every language is using its own specific stemming technique for increasing the efficiency.

D. Normalization

Normalization is simply the process of converting the texts in the corpus and query into the same case for preserving meaning. This is because most of the time terms may vary in their case regardless of their meaning and also, different users type their queries in different cases. For example, the Afaan Oromo terms "qoricha" to mean *medicine* is written as "Qoricha" at the beginning and, it is written as "qoricha" at the middle of a sentence while it has the same meaning. Normalization is one method for handling such differences. Finally, all the texts in the Afaan Oromo corpus will be converted into the most widely used case, lower case, except for the terms in the exception list. Case folding is easy in Afaan Oromo for instance **Maqaa** to mean *name* is written as similar to **maqaa.**

E. Term by Document Matrix Generation

Latent semantic indexing being a variant of vector space method, the documents as well as the queries are characterized mathematically as vectors in some vector space [12][19]. To generate the term by document matrix, a code is written that first declares a matrix with dimensions; number of unique terms represented the rows by number of unique documents represented the columns in the number of different concept collection for accessing and control the content of all the documents of the matrix in the entire collection. The entries in the cells of the matrix are initialized to 0. Then, a text file containing the list of all unique terms is created from the term-document matrix. The frequency of each term in each document is read from the text file generated automatically and substituted in the appropriate cell of the matrix that can be defined in the JAMA library.

3.3.2 The Numerical Data-Preprocessing/Term Weighting

The statistical weighting was done on by using tfidf weighting scheme. The TF is implemented using feature hashing for a better performance, since each raw feature is mapped into an index and IDF is implemented using hash for measuring how much information is given by a term according to its document frequency. From those only the body of the term by document matrix is generated in the term by document matrix. Then, the weighted matrix is imported into the SVD function. The basic rational behind weighting is that a term has high weight if it is frequent in the relevant documents but infrequent in the document collection [19].

In fact, a code for a SVD function that computes the weight (tfidf) of the term document matrix is written and import into JAMA library. The function accepts the raw term-document matrix as input

for SVD and applies the weighting to each of its elements and returns a weighted term documents matrix as value that show the significance of the term in the document collection used for index term. Here, the function is intentionally made to return the value that show the importance of the term in the document collection of each term, because, the values of the terms in the document are used later for weighting the query vectors.

3.4 K-dimensional Singular Value Decomposition (SVD)

The weighted term-document matrix "W" is computed in the previous stage through (tfidf) is used as an input for SVD and then, the SVD also computed for a reduced dimension K, where K is less than both the number of terms and the number of documents in the matrix. The K-dimensional SVD of the weighted matrix is performed by using SVD function in the JAMA library for computing the values. The SVD function accepts the weighted matrix and the number of dimensions as an input and returns three matrices, because it reduce the dimension of the data set and the three matrices are: the K-largest left singular vectors, Uk, of W, the K-largest right singular vectors, Vk, of W, and a diagonal matrix Sk whose non-zero entries in the principal diagonal are the singular values of W arranged in decreasing order and then the system checks if the user entered some value denoting the number of singular values. In simple way, [Uk, Sk, Vk]=svd(W, k), where W is the term-document weighted matrix and k is the reduced dimensionality of the weighted term-document matrix.

Choosing the Number of Dimensions

In the LSI there is no way to determining the optimal number of dimensions to use for performing the SVD. As we have mentioned in section 2.4.1, there is no general rule that can be used to select the optimal value of dimension K. So, in this thesis, the value for the dimension is chosen by the principle of 'What works best' and suitable depending on the nature of the document collection. That means, the retrieval performance of the LSI model was examined for several different dimensions and the dimensionality that maximizes the performance of the system was chosen.

3.5 Query Projection, Matching and Ranking of Relevant Documents.

3.5.1 Query Projection

The system takes user queries and then the queries are treated as pseudo-documents for the user. Therefore, the natural language preprocessing performed on the collected test document is, also performed on the queries and the vectors representing the frequency of each term in the queries are

attained. In addition to that, the query could represent in the same space with the collection test document, after that, they are weighted by using the tfidf weighted used in weighting the documents, and then projected into the space.

A SVD function, designed to compute the weighting and project the queries into the reduced space using the formula "**q=qT *Uk * Sk-1**" which is stated in section 2.4.2 (equ 2.12).

3.6.2 Matching and Ranking of Relevant documents.

When you find an LSI-indexed database, the queries and the documents are located in the same reduced LSI space, it is possible to compute the Euclidean distance between the queries and each document and take the nearest documents as the best ones for the specific query because ideally concept space is much less than the word space. However, as the term-document matrix was not normalized, a decision was made to use the cosine similarity measure, because by computing the angle, the lengths of the documents, which can affect the distance between the query and the documents in the space can be normalized [12][19].

Consequently, having the reduced dimensional representation of the query vector 'q' and the scaled document matrix'd' (i.e. d=Vk*Sk), the angle between them can be calculated by the formula.

$$\cos(x) = \frac{\vec{d} \cdot \vec{q}}{\left|\vec{d}\right|\left|\vec{q}\right|}$$

Where, ||q|| and ||d|| are the norm of the query and document vector respectively.

After the cosine measure is computed, the documents are sorted according to the cosine coefficients, the larger the cosine coefficient, the more relevant the document and LSI returns the relevant documents that do not contain the keyword at all.

3.6 Information Retrieval Evaluation System

Retrieving relevant document from the collection that satisfies users information need is the heart of Information retrieval evaluation system .Once the IR system is developed, it has to be tested with several queries before its final implementation. The testing process is the evaluation of the performance of the system. The type of evaluation to be considered depends on the objectives of the retrieval system [1]. The most common measures of system performance are efficiency (like time, space and corpus size). However, from an academic perspective, measurements are focused on the

specific effectiveness of a system and usually are applied to determining the effects of changing a system's algorithms or comparing algorithms among systems.

In a system designed for providing IR, other metrics, besides efficiency (like time, space and corpus size), are also of interest. In fact, since the user query request is inherently unclear, the retrieved documents are not exact answers and have to be ranked according to their relevance to the query. Thus, information retrieval systems require the evaluation of how precise the answer set is. This type of evaluation is known as retrieval performance evaluation [1]. Retrieval performance could be evaluated using different techniques such as recall, precision, E-measure, the harmonic mean (F-measure), Map (Mean Average Precision) and others. However, the most widely used retrieval performance measures are Recall and Precision. Recall and precision is the evaluation technique of the proposed information retrieval system. Recall is the percentage of relevant documents retrieved from the database in response to user's query, while Precision is the percentage of retrieved documents that are relevant to the query (i.e. number of retrieved documents that are relevant).

$$\text{Recall} = \frac{\text{No. of relevant documents retrieved}}{\text{Total no. of relevant documents in database}}$$

$$\text{Precision} = \frac{\text{No. of relevant documents retrieved}}{\text{Total number of documents retrieved}}$$

However, most of the time retrieval algorithms are evaluated by running them for several distinct queries. In this case, for each query a distinct precision versus recall curve is generated. To evaluate the retrieval performance of an algorithm over all test queries, we average the precision figures at each recall level.

Average precision versus recall figures are now a standard evaluation strategy for information retrieval systems and are used extensively in the information retrieval literature [1]. Hence, in this study the retrieval effectiveness of the system is evaluated using Average precision versus recall curve in the LSI dimension.

Chapter four
Experimentation and discussion

4.1. Introduction

When you discussed in chapter one, the main objective of this study is to experiment on the possibility of developed semantic-based Indexing technique for the retrieval of Afaan Oromo text corpus. Whether the proposed objective is achieved or not should be experimentally tested before concluding the possibility of developing such a system. Also the performance of the system should be tested that can be collected from the official website of Oromiya National Regional State, Voice of America Radio Afaan Oromo service, Gumii Waaqeffattoota Addunyaa (GWA) Portal, International Bible Society official website, Oromiya Radio and Television Organization (ORTO) news and other Internet based sources and the test result of the experimentation is also discussed in this chapter.

4.2 Description of the Prototype System.

The critical objective of this prototype model is to represent documents, queries and index terms in the similar reduced-dimensional space so that comparison of the documents, queries and index terms in this reduced-dimensional space is possible. In the middle of the prototype model, a standard vector space model is obtained. This makes the concept space is ideally much lesser than the word space because the LSI model straightforward.

The collection of the test document and the sample query files stored in their respective folders are read and each terms are extracted excluding the stop word. The term by document matrix and the query vector are generated from it. For the purpose of enhance or reduce the significance of each terms based on their capability to represent the content of a document and also distinguish it from other documents in the collection of the documents; both the term-document matrix and the query vectors are weighted.

The weight of the term-document can be calculated using ***tfidf*** term weighting methods. The term weighting was done by finding of the frequency of terms in the each document and its synonyms in the documents and product of its inverse document frequency of it. The weighted term-document matrix is then given to the SVD algorithm as an input and a reduced dimensional representation of the matrix is generated from it by calling the method to perform singular value decomposition. The

reduced-dimensional representation of the query vectors is also obtained through projection into the reduced space. After the documents, terms and queries are represented in the similar reduced-dimensional space, the cosine similarity measure is used to categorize the ranked list of relevant documents for each of the required query from a collection of documents.

The system has been built using Java; NetBeans 8.0.1 Figure 4.1, presents a screen shot which shows the first list of retrieved document using a given query

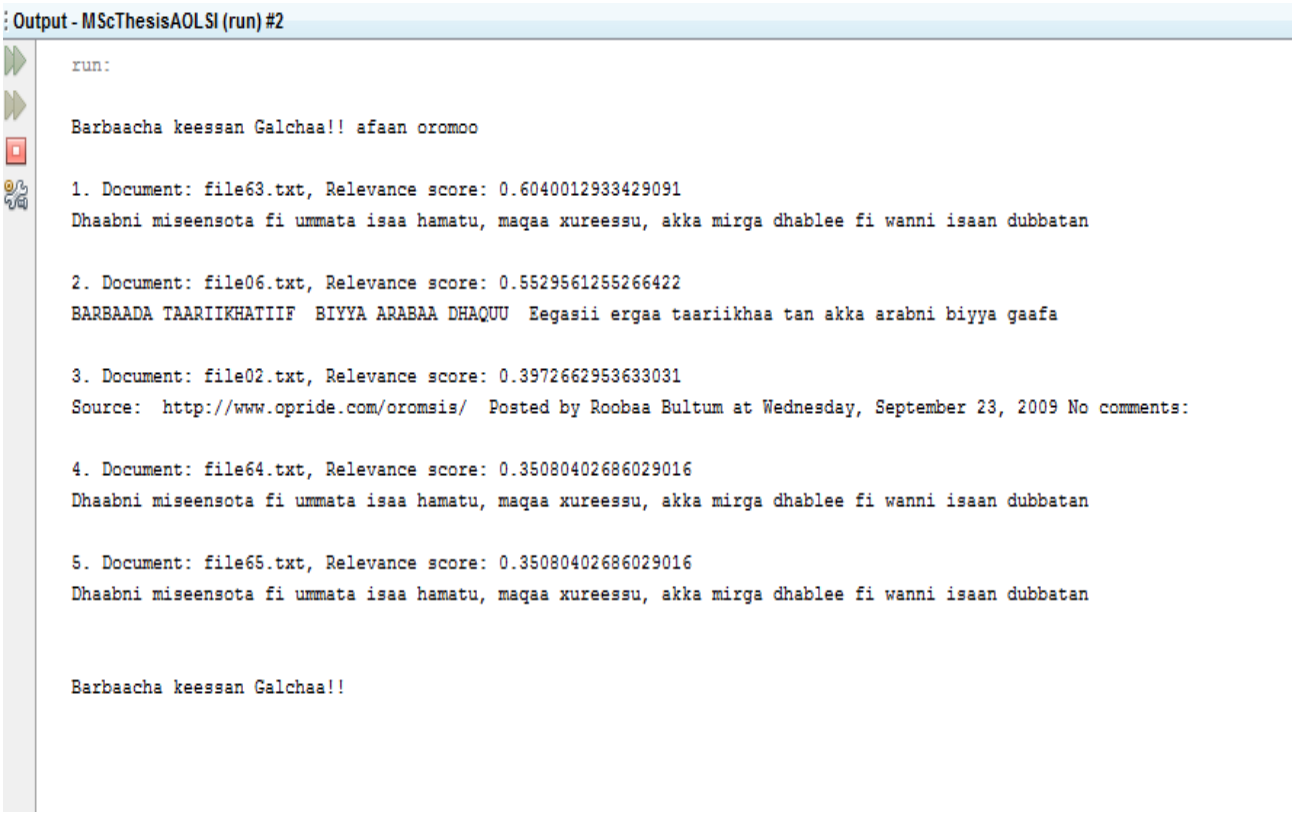


Figure 4.1: A Screen shot of retrieved document for a given query

Several preprocessing computation were applied on the collected documents to make them readily available to be being indexed and searched. After the document identified and collected the data preprocessing tasks are followed.

One of the major preprocessing tasks is tokenization. The tokenization of tagged documents in the corpus and query string is implemented in the Java; NetBeans 8.0.1. Tokenization in this work also used for splitting document in to tokens and detaching certain characters such as punctuation marks or non content bearing words[1]. Then convert every word in document in to lower case if it is not in

lower case primarily. The documents in lower case are checked as if it is not punctuation mark or number. Finally the tokenized document will be returned for next process.

After the tagged documents tokenized, the system extracts the content bearing terms like nouns and verbs of the document from the token set to keep the semantics of the document. On the semantics set of the token set some major activities carried out; stop word removal, stemming, and normalization.

It removes the stop words from the sub set of the token set, as a result token set would have only content bearing terms. The index terms selected in this study are content bearing terms which are not part of stop list. Primarily there are identified list of stop words which are not content bearing, just used for grammatical purpose only.

Afaan Oromo stop word list is saved in text file 'stopwordlist.txt'. Mostly, non-content bearing terms are removed by either removing high frequency terms from the indexing term list or by using a negative dictionary (stop word list) for that language[1][2]. Some of Afaan Oromo stop words are listed in Appendix I. First the algorithm reads the files and saves it on variable. Then normalized token is checked to be different from the terms in the stop word. Terms not stop word is forwarded to the stemmer function.

Stemming can be done for computational efficiency and better retrieval performance it is crucial to conflate morphological variations. Thus, stemming of the Afaan Oromo corpus and query is done by using a rule based stemmer developed by Debela T and Ermias A [35] which has six rules. Stemming techniques are language dependent. Therefore, every language is using its own specific stemming technique. So, the techniques are implemented in java.

Finally normalization performed because most of the time words may vary in their case regardless of their meaning and also, different users type their queries in different cases. For example, the Afaan Oromo word "qoricha" to mean *medicine* is written as "Qoricha" at the beginning and, it is written as "qoricha" at the middle of a sentence while it has the same meaning.

The next part is creating inverted file form by using term by document matrix generation latent semantic indexing being a variant of vector space method, the documents as well as the queries are represented mathematically as vectors in some vector space [12][19]. To generate the term by

document matrix, a code is written that first declares a matrix with dimensions; number of unique terms represented the rows by number of documents represented the columns in the collection that consists of methods and data structures to prepare and retrieve term frequency (TFIDF) values in the form of a double dimension array to allow construction of the term document matrix. The entries in the cells of the matrix are initialized to 0. Then, a text file containing the list of all individual terms is created. The frequency of each term in each document is read from the text file generated and replaced in the appropriate cell of the matrix class that can defied in the JAMA package.

Then the numerical data-pre-processing/weighting was done by using ***tfidf*** term weighting methods. The term weighting was done by finding of the frequency of terms and its synonyms in the documents and product of its inverse document frequency in the documents. Only the body of the term by document matrix generated in the term by document matrix generation, and saved into a text file in a folder. Then, the matrix is imported into the JAMA package. The basic rational behind weighting is that a term has high weight if it is frequent in the relevant documents but infrequent in the document collection [19]. Term weighting has been employed in the form of a double dimension array to allow building of the term document matrix.

In fact, a code for an inverted function that computes the weight of the term document matrix (tfidf) is written. The function accepts the raw term-document matrix as input apply the weighting to each of its elements and return a weighted term documents matrix as well as the weights value that indicate the importance of the term in the document collection used for index terms. Here, the function is intentionally made to return the values of weight importance of the term in the document collection of each term, because, the values are used later for weighting query vectors.

```
public class AfaOroIR {
TreeMap<String, Double> similarityValues;
   HashMap[] termfrequency;
   HashMap<String, Double> idf = new HashMap();
   HashMap<String, Double> tfidf[];
   List<String> all_terms = new ArrayList<String>();
   List<Double> all_tfidf = new ArrayList<Double>();
   List<Double> tdm = new ArrayList<Double>();
   List<String> terms = new ArrayList<String>();
   HashMap<String, Double> matrix = new HashMap<>();
   double termMatrix[][] = null;
   double[][] tfIdfMatrix;
   String filenames[] = null;
```

```
public static HashMap<String, Integer> stopwordlist = new HashMap<String, Integer>();

String delimiters = " \n\r\t.?!(),:;\"/\\[]{}?|@#*~$%^&<>_+=-...-_+£";
String digits = "0123456789";
int ndocs = 0;

void calculateFrequency() throws Exception {
   // public double idf(List<List<String>> docs, String term)
   FileReader stopwordfile = new FileReader("StopWordList.txt");
   BufferedReader stop = new BufferedReader(stopwordfile);
   String line = "";
   int j = 0;
   while ((line = stop.readLine()) != null) {
      stopwordlist.put(line.trim().toLowerCase(), j);
      j++;
   }

   File corpus = new File("corpus");
   filenames = corpus.list();
   //System.out.println(Arrays.toString(filenames));
   ndocs = filenames.length;
   int i = 0;
   termfrequency = new HashMap[filenames.length];
   while (i < filenames.length) {
      termfrequency[i] = calcFrequency(filenames[i]);
      //System.out.println(termfrequency.size());
      //System.out.println(filenames[i]);
        i++;
   }

   for (i = 0; i < filenames.length; i++) {
      System.out.println(filenames[i] + "--------------------" + termfrequency[i].size() + " distinct words:");
      //Prints the Number of Distict words found in the files read
      Iterator itr = termfrequency[i].keySet().iterator();
      while (itr.hasNext()) {
         String temp = itr.next().toString();
         //System.out.println(temp + " " + termfrequency[i].get(temp));

         terms.add(temp);
      }
   }
}

void calculateIDF() throws Exception {

   for (int i = 0; i < termfrequency.length; i++) {
      Iterator itr = termfrequency[i].keySet().iterator();
      while (itr.hasNext()) {
         String term = itr.next().toString();
         double idfv = getIDF(term);
         idf.put(term, idfv);
```

```
        }
    }
}

double getIDF(String term) {
    int docfreq = 0;
    for (int i = 0; i < termfrequency.length; i++) {
        HashMap hm = termfrequency[i];
        if (hm.containsKey(term)) {
            docfreq++;
        }
    }

    return Math.log10((double) ndocs / (double) docfreq) + 1;
}

void calculateTFIDF() throws Exception {
    System.out.println("Creating the tf-idf weight vectors");
    tfidf = new HashMap[termfrequency.length];
    for (int i = 0; i < termfrequency.length; i++) {
        HashMap<String, Double> temp = new HashMap<>();
        Iterator itr = termfrequency[i].keySet().iterator();
        while (itr.hasNext()) {
            String term = itr.next().toString();
            double idfv = idf.get(term);
            double tfidfv;
            tfidfv = (double) (idfv) * (int) termfrequency[i].get(term);
            //System.out.println(term + ": " + termfrequency[i].get(term) + ":" + idfv + "   :    " + tfidfv);
            temp.put(term, tfidfv);
            all_terms.add(term);
            all_tfidf.add(idfv);
            //System.out.println(temp.size());
        }
        //System.out.println(temp.size());
        tfidf[i] = temp;
    }
    for (int i = 0; i < all_terms.size(); i++) {
        matrix.put(all_terms.get(i), all_tfidf.get(i));
    }
     System.out.println(all_terms.size());
    //System.out.println(matrix);

}

// displaying term matrix
void displayMatrix() throws Exception {
    int a = all_terms.size();
    int b = filenames.length;
    termMatrix=new double[a][b];
    //  termMatrix[a][b]=null;
    for (int i = 0; i < all_terms.size(); i++) {
```

```
        //System.out.print(all_terms.get(i) + "     ");
        for (int j = 0; j < filenames.length; j++) {
           // System.out.print(all_terms.get(i));
           //if(filenames[i].)
            HashMap<String, Double> temp =tfidf[j];
           if (temp.containsKey(all_terms.get(i))) {
              System.out.print(temp.get(all_terms.get(i)) + "  ");
              //tdm.add((Double) temp.get(all_terms.get(i)));
              termMatrix[i][j] = temp.get(all_terms.get(i));
              System.out.println(termMatrix[i][j]+" ");
           }
           else {
              termMatrix[i][j] = 0;
                  System.out.print(0 + "                  ");

           }
        }
        System.out.print("\n");
     }
  }
```

The weighted term-document matrix ‘W’ of the preceding step is used as an input and its SVD is computed for a reduced dimension K, where K is less than both the number of terms and the number of documents in the matrix. The function accepts the weighted matrix and the number of dimensions as an input and returns three matrices: the K-largest left singular vectors, K-largest right singular vectors, and a diagonal matrix Sk whose non-zero entries in the principal diagonal are the singular values of Weight arranged in decreasing order and then the system checks if the user entered some value denoting the number of singular values.

There is no general rule for choosing the number of the optimal value of K dimension. In this study, the value for the dimension is chosen by the principle of ‘What works best’. That means, the retrieval performance of the LSI model was examined for several different dimensions and the dimensionality that maximizes performance was preferred.

The system takes user queries and then the queries are treated as pseudo-documents. So, the natural language preprocessing achieved on the collection of the test document is also performed on the queries and the vectors specifying the frequency of each term in the queries are obtained. In addition to that, in order to represent the queries in the same space with the test document collection, they are

weighted using the same scheme used in weighting the documents, **tfidf**, and then projected into the same space.

A SVD function, designed to perform the weighting and project the queries into the reduced space using the formula "**q=qT *Uk * Sk-1**" which is stated in section 2.4.2 (equ 2.12).

Now, that the queries and the documents are located in the same reduced LSI space. Because the concept space is ideally much lesser than the word space, we can limit the number of singular values such that important semantic information is not lost and noise is removed from the system. To handle the query entered by the user and then compute similarity of the query against documents in the corpus using Latent Semantic Indexing. Consequently, having the reduced dimensional representation of the query vector 'q' and the scaled document matrix'd', the angle between them can be computed by cosine similarity formula.

$$\cos(x) = \frac{\vec{d} \cdot \vec{q}}{\left|\vec{d}\right|\left|\vec{q}\right|}$$

Where, ||q|| and ||d|| are the norm of the query and document vector respectively.

After the cosine measure is computed, the documents are sorted according to the cosine value, the larger the cosine value, the more relevant the document to the user query using the concept of the LSI approaches.

4.3 Performance Evaluation

As discussed in section 3.6, Precision, Recall and F-measure are the most frequent and basic statistical measures which widely used to measure the effectiveness of IR systems (i.e., the quality of the search results). The two parameters (Precision and Recall) are used in this research so as to measure the effectiveness of the designed Latent Semantic Indexing based IR system.

4.4 Experimentation

In this experiment nine (9) queries are used for evaluating the performance of the system. A result of each query is presented in term of precision and recall in Table 4.2.

Query number	List of Query terms	Relevant document retrieved	List of retrieved relevant documents for queries

Query1	*Afaan Oromoo*	*file63.txt,File62.txt, file06.txt, file02.txt,file64.txt file65.txt, file36.txt, file02.txt*	*file63.txt, file06.txt, file02.txt,file64.txt ,file65.txt*
Query2	*Mirga dhala nama*	*File15.txt, File16.txt, file44.txt,file70.txt file22.txt, File41.txt, File291.txt, File30.txt,*	*File15.txt, file44.txt, file37.txt,file70.txt file22.txt*
Query3	*Sirna Gadaa*	*File10.txt,File01.txt,File23.txt,file08.txt,file27.txt File24.txt,file16.txt,file11.txt file23.txt*	*File01.txt, file16.txt,file11.txt file23.txt*
Query4	Oomisha xaafii Oromiyaa	*File08.txt, file16.txt, file07.txt,file01.txt*	*File08.txt, file16.txt, file07.txt, file10.txt, File20.txt,*
Query5	*Makmmaakssa oromoo*	*File63.txt, File62.txt, file64.txt,*	*File63.txt,file64.txt,file65.txt, file36.txt,file02.txt,file62.txt,file36.txt file02.txt , File11.txt,*
Query6	*Tapha kubbaa miila* Itiyoophiyaa	*File03.txt, file04.txt,file10.txt,file51.txt, file55.txt*	*File03.txt,file04.txt, file10.txt,file51.txt file55.txt, File01.txt,File02.txt,File57.txt File59.txt*
Query7	*Oduu*	*File42.txt,File43.txt,File45.txt,File01.txt, file04.txt, file44.txt,file36.txt file35.txt, File37.txt,*	*File01.txt, file04.txt, file44.txt, File42.txt,File43.txt,File45.txt*
Query8	*Ayyaana Irrechaa*	*File04.txt, file58.txt*	*File04.txt, file59.txt, file44.txt,file42.txt file09.txt*
Query9	*Bilisumma ummata oromoo*	*File63.txt,file64.txt,file65.txt,file02.txt,file06.txt*	*File63.txt,file64.txt, file65.txt,file02.txt,File01.txt, File03.txt,File05.txt*

Table 4.2. frequent string query, relevant documents and retrieved document by prototype system

Based on the information given in above Table 4.2 the recall and precision were calculated for each query, they were calculated using equation that can listed in section 3.6. Table 4.3 shown the effectiveness of latent semantic indexing for Afaan Oromoo IR system based the above selected queries for the experimenting.

Query terms	**Relevant**	**Retrieved**	**Relevant retrieved**	**Recall**	***Precision***
Afaan Oromoo	*8*	*5*	*5*	*0.63*	*1.00*
Mirga dhala nama	*5*	*8*	*4*	*0.80*	*0.50*
Sirna Gadaa	*4*	*9*	*3*	*0.75*	*0.33*
Oomisha xaafii Oromiyaa	*5*	*4*	*3*	*0.60*	*0.75*
Makmmaakssa oromoo	*9*	*3*	*3*	*0.33*	*1.00*
Tapha kubbaa miila Itiyoophiyaa	*5*	*9*	*5*	*1.00*	*0.55*
Oduu	*6*	*9*	*5*	*0.83*	*0.56*
Ayyaana Irrechaa	*5*	*2*	*1*	*0.20*	*0.50*
Bilisumma ummata oromoo	*7*	*5*	*4*	*0.57*	*0.80*
Average				***0.63***	***0.67***

Table 4.3 Detailed result of the performance achieved

As it is shown in Table 4.4 the obtained result is 0.67(67%) precision and 0.63(63%) recall. The performance of the system is lowered for various reasons as it is identified by this study. Precision evaluates that all relevant document are retrieved. In this study the systems only retrieves 67% relevant documents for these given nine (9) queries.

In the above tables shown that there were irrelevant documents retrieved from the collection of documents (corpus) and some of the relevant document from the corpus could not be retrieved. For example, when we tried to search the relevant document from the corpus using "Afaan Oromoo" query. we obtained better precision(*1.00*) but lower recall(*0.63*). In this case all relevant documents

are retrieved but there irrelevant documents which are retrieved too. This is because the term "Afaan" has many meaning and found in different documents.

When you saw, the values of precision at standard recall levels for collection of query were important so as to show the performance of the system.

Thus, the interpolated recall-precision curve to the performance of the designed prototype system is done in the figure.

As the curve shown when the recall increase the precision decrease and vice versa. It occurs because one enhances with the cost of the other.

In general the work done is recorded promising performance. Better performance will be achieved if there is mechanism to partially handle polysemy and synonymy by using LSI.

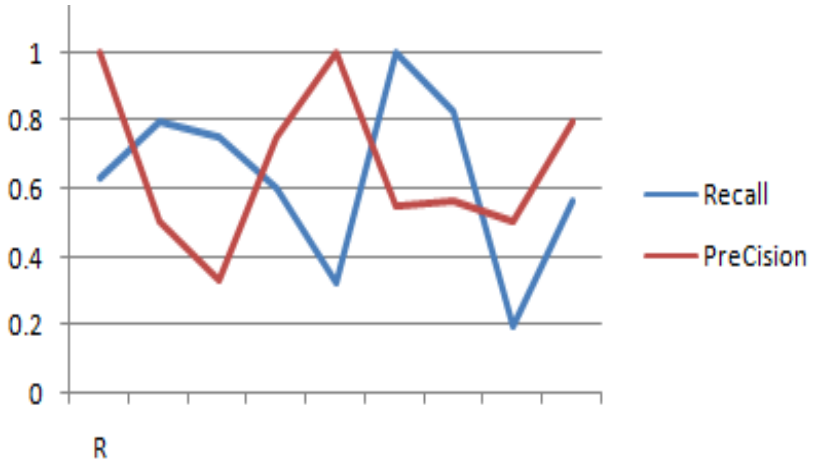


Figure: 4.1 recall versus precision interpolation

Chapter five
Conclusion and recommendations

5.1. Introduction

Throughout this research an effort has been made to develop Applications of Information Retrieval for Afaan Oromo text based on Semantic-based Indexing language. These chapters have two main parts, Conclusion and Recommendation. The first part gives the concluding points of the results obtained in the course of this research work while the second part discusses directions for future research.

5.2. Conclusion

Information retrieval (IR) is concerned with locating documents that are relevant for a user's information need or query from a large collection of documents. A fundamental problem with the classic information retrieval systems is that different ways of expressing the same concept, the terms in the user's query may not appear in a relevant document and many words can have more than one meaning. A query is often a short and incomplete description of the information need. The users of IR systems and the authors of the documents often use different words to refer to the same concept. Due to these facts term matching methods are likely to miss relevant documents and also retrieve irrelevant ones.

The goal of the research is to examine the benefits of LSI text retrieval approach for Afaan Oromoo text retrieval. LSI attempts to use this as a basis for creating a concept space such that this concept space is much smaller than the word space (mapping of terms against documents). The initial concept space consists of mapping of all terms against all documents in the form of a matrix. This matrix is then subjected to singular value decomposition. The weighted matrix was analyzed further by the Singular Value Decomposition to derive the latent structure model which was used for indexing and retrieval by the LSI retrieval model The matrices thus received are then used to represent the relationships between terms and documents. Two documents (of which one can be a query) can then be compared by comparing the concepts that they represent. For these reason, it uses concepts or topics instead of individual words to index and retrieve the documents, consequently allowing a relevant document to be retrieved even when it shares no common words with in the query [1][2]. Investigation on the retrieval performance of two text retrieval approaches has been

made: standard vector space; and Latent Semantic Indexing, and the experimental results have been presented.

In this study, a test collection of 70 news articles were taken. The whole text of the news was used for indexing purpose. At some point in the preparation of the text for indexing, punctuation marks, space, tab, carriage return and line feed character were used as word identifiers.

Each document is indexed automatically. All terms that don't occur in stop word list were included in the study. The automatic indexing resulted in a total 8625 unique words.

The analysis begins with a weighted term by document matrix in which each number in the cell of the matrix represent the importance of the term within as well as in the entire document collection.

Cosine similarity measure between the query vector and document vectors were used to identify documents which are near (relevant) to each queries and the documents are ranked according the their cosine measures.

At last, the retrieval model used for the study is latent semantic indexing (LSI) method. According to the experimentation made the system registered 0.67 precision and 0.63 recalls which is promising to Applications of Information Retrieval for Afaan Oromo text based on Semantic-based Indexing system that searches within large text corpus.

5.3 Recommendation

The following recommendations forwarded based on the finding which include the developments of resources and future research directions to Applications of Information Retrieval for Afaan Oromo text based on Semantic-based Indexing:

Future research directions for Applications of Information Retrieval for Afaan Oromo text based on Semantic-based Indexing: include:

- Applying LSI for cross language retrieval (Afaan Oromoo vs. English) and information filtering on Afaan Oromoo documents is highly advisable.
- This work is a text retrieval system for under resourced languages like Afaan Oromoo documents. Other types of document like video, audio, graphics and pictures are not studied. It is very important to make more study to come up integrated and fully functional system.
- Information retrieval needs standard corpus for testing and making experimentation. But there is no standard corpus developed yet. This needs great emphasis as future work.

Reference

[1] R. Baeza-Yates and B. Ribeiro-Neto, Modern information retrieval .ACM press New York, 1999.

[2] C. D. Manning, P. Raghavan, and H. Schutze, "An Introduction to Information Retrieval, Online Edition," Cambridge: Cambridge UP, 2009,

http://nlp.stanford.edu/IR-book/pdf/irbookonlinereading.pdf

[3] Amit Singhal.(2008) Modern Information Retrieval: A Brief Overview, Google, Inc, accessed June 2015, http://ilps.science.uva.nl/Teaching/0405/AR/part2/ir-overview.Pdf

[4] Bin H. Automatic Term Extraction in Large Text Corpora, Faculty of Computer Science, Dalhousie University, Halifax, Canada B3H 1W5

[5] Hans F.. Terminology Extraction and Automatic Indexing: Comparison and Qualitative Evaluation of Methods, http://wortschatz.uni-leipzig.de/~fwitschel/papers/TKEIndexing.pdf

[6] "Summary and Statistical Report of the 2007 Housing Census: Population Size by Age and Sex", Addis Ababa, 2008.

[7] Linda A., Performance of Two Statistical Indexing Methods, with and without Compound-word Analysis,

[8] W. Bruce C.. (1995), "What Do People Want from Information Retrieval?", accessed 11 February 2013, http://www.dlib.org/dlib/november95/11croft.html

[9] Tilahun, G. Qube Afaan Orom: Reasons for Choosing the Latin Script for Developing an Oromo Alphabet, The Journal of Oromo Studies, 1993.

[10] Assefa W/mariam, Developing Morphological Analysis for Afaan Oromo Text, Master Thesis, School of Graduate studies, Addis Ababa University, 2005.

[11] Abara, N. Long Vowels in Afaan Oromo: A Generative Approach, Master Thesis, School of Graduate Studies, Addis Ababa University, 1988

[12]Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, Richard Harshman, Indexing by Latent Semantic Analysis.

[13] Barbara R. Latent Semantic Indexing: An overview, INFOSYS 240 Spring 2000, Final Paper.

[14] Ashwini Deshmukh, Gayatri Hegde, A literature survey on latent semantic indexing, Computer Engineering Department, Mumbai University, New Panvel, India. 2012

[15] Gezehagn Gutema Eggi, "Afaan Oromo Text Retrieval System",MSc Thesis, School of Information Science, Addis Ababa University, 2012.

[16]. Daniel Bekele Ayana, "afaan oromo-english cross-lingual information retrieval (clir): a corpus based approach", MSc Thesis, School of Information Science, Addis Ababa University, 2011.

[17] Aurélie N., James G., and Alan R.n**.** Automatic Indexing of Specialized Documents: Using Generic vs. Domain-Specific Document Representations, National Library of Medicine, 8600 Rockville Pike, Bethesda, MD 20894, USA.

[18] I. Jukes, "From Gutenberg to Gates to Google and Beyond: Education For The On-Line World", Singapore, pp. 1-144, 2005.

[19] Dumais, S. T. "Enhancing Performance in Latent Semantic Indexing (LSI) Retrieval", 1992.

[20] Gezahegn G., Afaan Oromo text retrieval system, MSc Thesis, School of Information Science, Addis Ababa University, Ethiopia, 2012

[21] Tabor Wami's, new book titled, Yewugena Dirsetoch ena Yetarik Ewunetawoch, 2004, http://www.omniglot.com/writing/oromo.htm last accessed Wednesday, November 18, 2015.

[22] Taha, R. Modern Afaan Oromo Grammar, ISBN: 9781468515060, 2004.

[23] Gumii Qormaata Afaan Oromoo, Caasluga Afaan Oromo, Jildi I, Komishinii Aadaaf Turizmii Oromiyaa, Finfinnee, 1995.

[24] Tewodros H., Amharic text retrieval: an experiment using Latent semantic indexing (LSI) with singular value decomposing (SVD), MSc thesis, School of Information Science, Addis Ababa University, Ethiopia,2003.

[25] Abey B., Semantic Based Query Expansion Technique for Amharic IR, MSc Thesis, School of Information Science, Addis Ababa University, Ethiopia, 2011

[26] Mulualem W., Semantic Indexing and Document Clustering For Amharic Information Retrieval, MSc Thesis, School of Information Science, Addis Ababa University, Ethiopia, 2013

[27] Daniel Bekele, Afaan Oromo-English Cross-Lingual Information Retrieval (CLIR): A corpus based approach, MSc Thesis, School of Information Science, Addis Ababa University, 2011.

[28] Eyob N., Afaan Oromo-Amharic Cross-Lingual Information Retrieval (CLIR): A corpus based approach, MSc Thesis, School of Information Science, Addis Ababa University, 2013.

[29] Amanuel Raga and Samuel Adola, Homonymy as a barrier to mutual intelligibility among speakers of various dialects of Afaan Oromo,Journal of Language and Culture, vol. 3, no. 2, pp. 32-43, 2012.

[30] Debela T, Designing a Stemmer for Afaan Oromo Text: A hybrid approach, MSc Thesis, School of Information Science, Addis Ababa University, 2010.

[31] Diriba M, An automatic sentence parser for Oromo language using supervised learning techniques**,** MSc Thesis, School of Information Science, Addis Ababa University, 2002.

[32] Meiws C.G, A grammatical sketch of Written Oromo, ISBN 3- 89645- 039-5, 2001.

[33] Amanuel R and Samuel A, Homonymy as a barrier to mutual intelligibility among speakers of various dialects of Afaan Oromo, Journal of Language and Culture ,Vol. 3(2), pp. 32-43,2012

[34] Chih-Ping Wei, Christopher C. Yang, Chia-Min Lin, A Latent Semantic Indexing-based approach to multilingual document clustering, 2007.

[35] Tesfaye D and Ermias A, Designing a Rule Based Stemmer for Afaan Oromo Text,‖ International journal of Computational Linguistics, Addis Ababa, vol. 1, no. 2, pp. 1-11, 2010.

[36] Bo-Yeong K, A Novel Approach to Semantic Indexing Based on Concept, Department of Computer Engineering, Kyungpook National University

[37] G.bharathi and D.venkatesan. , improving information retrieval using document clustering and semantic synonyms extraction, School of Computing, SASTRA University, Tamil Nadu, India.

[38] Teferi Kebebew, Speech recognition for Afaan Oromoo using Hybrid Hidden Markov Models and Artificial Neural Network, MSc Thesis, School of Information Science, Addis Ababa University, 2016.

[39] Yehuwalashet Bekele Tesema, Hybrid Word Sense Disambiguation Approach for Afaan Oromo Words, MSc Thesis, School of Information Science, Addis Ababa University, 2016.

Appendix I Afaan Oromo Stop word list (EGGI, 2012)

anee	fagaatee	isaanirraa	kan
agarsiisoo	fi	isaanitti	kana
akka	fullee	isaatiin	kanaa
akkam	fuullee	isarraa	kanaaf
akkasumas	gajjallaa	isatti	kanaafi
akkum	gama	isee	kanaafi
akkuma	gararraa	iseen	kanaafuu
ala	garas	ishee	kanaan
alatti	garuu	ishii	kanaatti
alla	giddu	ishiif	karaa
amma	gidduu	ishiin	kee
ammo	gubbaa	ishiirraa	keenna
ammoo	ha	ishiitti	keenya
an	hamma	ishiitti	keessa
ana	hanga henna	isii	keessan
ani	hoggaa	isiin	keessatti
ati	hogguu	isin	kiyya
bira	hoo	isini	koo
booda	hoo	isinii	kun
boooddee	illee	isiniif	lafa
dabalatees	immoo	isiniin	lama
dhaan	ini	isinirraa	malee
dudduuba	innaa	isinitti	manna

dugda	inni	ittaanee	maqaa
dura	irra	itti	moo
duuba	irraa	itumallee	na
eega	irraan	ituu	naa
eegana	isa	ituullee	naaf
eegasii	isaa	jala	naan
ennaa	isaaf	jara	naannoo
erga	isaan	jechaan	narraa
ergii	isaani	jechoota	natti
f	isaanii	jechuu	nu
faallaa	isaaniitiin	jechuun	nu'i
nurraa	saaniif	tahullee	waan
nuti	sadii	tana	waggaa
nutti	sana	tanaaf	wajjin
nuu	saniif	tanaafi	warra
nuuf	si	tanaafuu	woo
nuun	sii	ta'ullee	yammuu
nuy	siif	ta'uyyu	yemmuu
odoo	siin	ta'uyyuu	yeroo
ofii	silaa	tawullee	yommii
oggaa	silaa	teenya	yommuu
oo	simmoo	teessan	yoo
osoo	sinitti	tiyya	yookaan
otoo	siqee	too	yookiin
otumallee	sirraa	tti	yookiinimoo

otuu otuullee	sitti sun	utuu waa'ee	yoom

***Appendix II*: Implementation source code in Java**

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package afaoroir;

public class AfaOroIR {
//read all documents and store on HashMap
//read all stopwordlist and store on HashMap
TreeMap<String, Double> similarityValues;
  // AfaanOromooInvertedFile afaanOroinvObj;
   int relevantDocs = 5;
   HashMap[] termfrequency;
   HashMap<String, Double> idf = new HashMap();
   HashMap<String, Double> tfidf[];
   List<String> all_terms = new ArrayList<String>();
   List<Double> all_tfidf = new ArrayList<Double>();
   List<Double> tdm = new ArrayList<Double>();
   List<String> terms = new ArrayList<String>();
   HashMap<String, Double> matrix = new HashMap<>();
```

```
double termMatrix[][] = null;
double[][] tfIdfMatrix;
String filenames[] = null;
Matrix TermDocs, leftSingularMatrix, rightSingularMatrix, singularValueMatrix;
SingularValueDecomposition svd;
int k;
boolean eligibleQuery = false;
//public static HashMap<String, Integer> documentterms = new HashMap<String, Integer>();
public static HashMap<String, Integer> stopwordlist = new HashMap<String, Integer>();

//public static List<Object> document = new ArrayList<>();
String delimiters = " \n\r\t.?!(),:;\"/\\[]{}?|@#*~$%^&<>_+=-...-_+£";
String digits = "0123456789";
int ndocs = 0;

//FileWriter writer = null;
// Scanner scanner = null;
//void calculateFrequency(List<String> doc,String term)
void calculateFrequency() throws Exception {
   // public double idf(List<List<String>> docs, String term)
   FileReader stopwordfile = new FileReader("StopWordList.txt");
   BufferedReader stop = new BufferedReader(stopwordfile);
   String line = "";
   int j = 0;
   while ((line = stop.readLine()) != null) {
```

```
            stopwordlist.put(line.trim().toLowerCase(), j);
            j++;
        }

        File corpus = new File("corpus");
        filenames = corpus.list();
        //System.out.println(Arrays.toString(filenames));
        ndocs = filenames.length;
        int i = 0;
        termfrequency = new HashMap[filenames.length];
        while (i < filenames.length) {
            termfrequency[i] = calcFrequency(filenames[i]);
            //System.out.println(termfrequency.size());
            //System.out.println(filenames[i]);
              i++;
        }

        for (i = 0; i < filenames.length; i++) {
            System.out.println(filenames[i] + "----------" + termfrequency[i].size() + " distinct words:");
            //Prints the Number of Distict words found in the files read
            Iterator itr = termfrequency[i].keySet().iterator();
            while (itr.hasNext()) {
                String temp = itr.next().toString();
                //System.out.println(temp + " " + termfrequency[i].get(temp));
```

```
            terms.add(temp);
        }
    }
}

void calculateIDF() throws Exception {

    for (int i = 0; i < termfrequency.length; i++) {
        Iterator itr = termfrequency[i].keySet().iterator();
        while (itr.hasNext()) {
            String term = itr.next().toString();
            double idfv = getIDF(term);
            idf.put(term, idfv);
        }
    }
}

double getIDF(String term) {
    int docfreq = 0;
    for (int i = 0; i < termfrequency.length; i++) {
        HashMap hm = termfrequency[i];
        if (hm.containsKey(term)) {
            docfreq++;
        }
```

```
        }

        return Math.log10((double) ndocs / (double) docfreq) + 1;
    }

    void calculateTFIDF() throws Exception {
        System.out.println("Creating the tf-idf weight vectors");
        tfidf = new HashMap[termfrequency.length];
        for (int i = 0; i < termfrequency.length; i++) {
            HashMap<String, Double> temp = new HashMap<>();
            Iterator itr = termfrequency[i].keySet().iterator();
            while (itr.hasNext()) {
                String term = itr.next().toString();
                double idfv = idf.get(term);
                double tfidfv;
                tfidfv = (double) (idfv) * (int) termfrequency[i].get(term);
                //System.out.println(term + ": " + termfrequency[i].get(term) + ":" + idfv + "    :     " +
tfidfv);
                temp.put(term, tfidfv);
                all_terms.add(term);
                all_tfidf.add(idfv);
                //System.out.println(temp.size());
            }
            //System.out.println(temp.size());
            tfidf[i] = temp;
```

```
    }
    for (int i = 0; i < all_terms.size(); i++) {
        matrix.put(all_terms.get(i), all_tfidf.get(i));
    }
     System.out.println(all_terms.size());
    //System.out.println(matrix);

}

// displaying term matrix
public double[][] getdisplayMatrixValues() throws Exception {
    //double[][] termMatrix = new double[all_terms.size()][filenames.length];
    int a = all_terms.size();
    //System.out.println(a);
    int b = filenames.length;
    termMatrix = new double[a][b];
    // System.out.println(b);
    //  termMatrix[a][b]=null;
    for (int i = 0; i < all_terms.size(); i++) {
        //System.out.print(all_terms.get(i) + "     ");
        for (int j = 0; j < filenames.length; j++) {
            // System.out.print(all_terms.get(i));
            //if(filenames[i].)
            HashMap<String, Double> temp = tfidf[j];
            if (temp.containsKey(all_terms.get(i))) {
```

```
                System.out.print(temp.get(all_terms.get(i)) + "  ");
                //tdm.add((Double) temp.get(all_terms.get(i)));
                termMatrix[i][j] = temp.get(all_terms.get(i));
                System.out.println(termMatrix[i][j]+" ");
            } else {
                termMatrix[i][j] = 0;
                    System.out.print(0 + "                    ");

            }
          }
           System.out.print("\n");
       }
       return termMatrix;
   }
   // createTermDocumentMatrix()  retrieves TF*IDF values in the form of a double dimension array where rows //represent terms and columnsrepresent documents
    void createTermDocumentMatrix() throws Exception{
       TermDocs = new Matrix(termMatrix);
       //System.out.println(TermDocs);
   }
 //Set the matrices as per value of k
   private void prepareMatrices() {
       for (int i = 0; i < leftSingularMatrix.getRowDimension(); i++) {
           for (int j = k; j < leftSingularMatrix.getColumnDimension(); j++) {
               leftSingularMatrix.set(i, j, 0);
```

```
        }
    }
    for (int i = k; i < singularValueMatrix.getRowDimension(); i++) {
        for (int j = k; j < singularValueMatrix.getColumnDimension(); j++) {
            singularValueMatrix.set(i, j, 0);
        }
    }

    for (int i = 0; i < rightSingularMatrix.getRowDimension(); i++) {
        for (int j = k; j < rightSingularMatrix.getColumnDimension(); j++) {
            rightSingularMatrix.set(i, j, 0);
        }
    }
}
```

//Prepares a object of class SingularValueDecomposition by passing the term document matrix TermDocs.

```
private void performSingularValueDecomposition() {
    svd = new SingularValueDecomposition(TermDocs);
    leftSingularMatrix = svd.getU();
    singularValueMatrix = svd.getS();
    rightSingularMatrix = svd.getV();

    if (k == -1) {
        k = leftSingularMatrix.getColumnDimension();
```

```
    }

    if (k >= 0 && k < leftSingularMatrix.getColumnDimension()) {
        prepareMatrices();
    }
}
private Matrix createQueryVector(String query) {
    List<String> words = Arrays.asList(query.split(" "));
    String wordList = tdm.toString();
    Matrix queryVector = new Matrix(termfrequency.length, 1);
    int termCounter;

    for (termCounter = 0; termCounter < words.size(); termCounter++) {
        words.set(termCounter, words.get(termCounter).toLowerCase());
    }

    termCounter = 0;
Object term = null;
  //       for (String term : wordList.freqhash {
        if (words.contains(term)) {
            queryVector.set(termCounter, 0, queryVector.get(termCounter, 0) + 1);
            eligibleQuery = true;
        }
        termCounter++;
  // }
```

```
    return queryVector;
}
public double getVectorModulus(Matrix matrix) {
    double product = 0;

    for (int i = 0; i < matrix.getRowDimension(); i++) {
        product += matrix.get(i, 0) * matrix.get(i, 0);
    }

    product = Math.sqrt(product);
    return product;
}
private double getModulus(Matrix matrix) {
    return (getVectorModulus(leftSingularMatrix.transpose().times(matrix)));
}
private Matrix getDocumentColumnMatrix(int columnNumber) {
    Matrix columnMatrix = new Matrix(rightSingularMatrix.getColumnDimension(), 1);
    Matrix resultMatrix;
    for (int i = 0; i < columnMatrix.getRowDimension(); i++) {
        columnMatrix.set(i, 0, rightSingularMatrix.get(columnNumber, i));
    }

    resultMatrix = leftSingularMatrix.times(singularValueMatrix);
    resultMatrix = resultMatrix.times(columnMatrix);
```

```
    return resultMatrix;
}
private double getSimilarity(Matrix queryMatrix, Matrix documentMatrix) {
    Matrix productMatrix;
    double denominator, similarity;

    productMatrix = (queryMatrix.transpose()).times(documentMatrix);
    denominator = getModulus(queryMatrix) * getModulus(documentMatrix);

    similarity = productMatrix.det() / denominator;
    return similarity;
}
private TreeMap<String, Double> findSimilarities(Matrix queryVector) {
    String[] documents = filenames;
    similarityValues = new TreeMap<String, Double>();
    Matrix documentMatrix;
    int columnNumber = 0;

    for (String document : documents) {
        documentMatrix = getDocumentColumnMatrix(columnNumber++);
        similarityValues.put(document, getSimilarity(queryVector, documentMatrix));
    }

    return similarityValues;
```

```
    }
    private String findMax(TreeMap<String, Double> similarityValues) {
        double max = Double.NEGATIVE_INFINITY;
        String maxDocName = null;
        for (String document : similarityValues.keySet()) {
            if (similarityValues.get(document) != Double.NaN && similarityValues.get(document) > max) {
                maxDocName = document;
                max = similarityValues.get(document);
            }
        }

        return maxDocName;
    }
    private void displayRelevantDocuments(TreeMap<String, Double> similarityValues, int docCount) throws IOException {
        if (docCount > TermDocs.getColumnDimension()) {
            docCount = TermDocs.getColumnDimension();
        }

        System.out.println();
        for (int i = 0; i < docCount; i++) {
            String documentName = findMax(similarityValues);
            if (documentName != null) {
                System.out.println(i + 1 + ". Document: " + documentName + ", Relevance score: " + similarityValues.get(documentName));
```

```
            //termMatrix.(documentName);
            similarityValues.remove(documentName);
        } else {
            System.out.println("\nOnly " + i + " relevant documents  there\n");
            return;
        }
    }
}
public void handleQuery(String query) throws IOException {
    Matrix queryVector = createQueryVector(query);
    if (eligibleQuery == true) {
        TreeMap<String, Double> documentSimilarityValues = findSimilarities(queryVector);
        displayRelevantDocuments(documentSimilarityValues, relevantDocs);
        eligibleQuery = false;
    } else {
        System.out.println("barbachii isin galchitan hundii stoopi wordii listii kessa jira ykn immoo jechii isin galchitan documenti hundaa kessa hin jiru");
    }
}
    boolean containsNumbers(String str) {
    for (int i = 0; i < str.length(); i++) {
        char ch = str.charAt(i);
        if (digits.contains("" + ch)) {
            return true;
        }
```

```
    }
    return false;
}

HashMap calcFrequency(String filename) throws Exception {
    HashMap<String, Integer> freqhash = new HashMap<String, Integer>();
    FileReader file1 = new FileReader("corpus/" + filename);
    BufferedReader bf = new BufferedReader(file1);
    String fline = "";
    while ((fline = bf.readLine()) != null) {
        fline = fline.toLowerCase();
        String string = "";
        StringTokenizer stok = new StringTokenizer(fline, delimiters);
        while (stok.hasMoreTokens()) {
            String word = stok.nextToken();
            if (containsNumbers(word)) {
                continue;
            }

            if (!stopwordlist.containsKey(word)) {
                int i = 0;
                if (freqhash.containsKey(word)) {
                    i = freqhash.get(word);
                    i++;
                    freqhash.put(word, i);
```

```
                } else {
                    freqhash.put(word, 1);
                }
                i++;
            }
        }

    }

    return freqhash;
  }

  public static void main(String[] args) throws Exception {
//
//
  }
}
```

Appendix III: Question for Collection of Vague Words

The questions are prepared and transformed to Afaan Oromo language to make suitable to gather the vague words test from public.

Saala: _____ Umurii:_____

Sadarka Barumsa: kutaa 1-8:_____

kutaa 9-12:_____

Barataa Kolleejjii:_____

Barataa Yuuniversitii:_____

Barsiisaa:_____

Qoottee Bulaa:_____

Kan biraa:_____

1. Jechoota hiika lamaa ol qaban (vague words) kan naannoo keessanitti fayyadamtu naaf barreessi?

Lakkofsa	jechoota	Baay'ina Hiikaa	hiika