ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
SCHOOL OF INFORMATION SCIENCE

Afaan Oromo Text Retrieval System

Gezehagn Gutema Eggi

June 2012
Addis Ababa, Ethiopia

ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
SCHOOL OF INFORMATION SCIENCE

Afaan Oromo Text Retrieval System

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF MASTER OF INFORMATION SCIENCE

Gezehagn Gutema Eggi

June 2012
Addis Ababa, Ethiopia

ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
SCHOOL OF INFORMATION SCIENCE

Afaan Oromo Text Retrieval System

Gezehagn Gutema Eggi

Signature of the Board of the Examiners for Approval

_____	_____	_____
_____	_____	_____
_____	_____	_____

This work is dedicated to my mom for she is the one who directed me this way!

Acknowledgements

The one above all, the omnipresent God, for answering my prayers for giving me the strength to plod on despite of my weakness wanting to give up and throw in the towel, thank you so much Dear Lord.

This thesis would not have become real without my advisor Dr Million Meshesha who's never gave up on helping me, from the beginning to end of the study. I am very grateful for his continuous help.

It is an honor for me to express my special appreciation for my classmates for their collaboration with giving me ideas, directions, comments and also for their encouragements.

It is a pleasure to thank those who helped me by different mechanism when I was working on this study specially Dr Jean Pierre Ilboudo and other peoples at UNESCO, my family and others people those their name is not mentioned here.

Abstract

This study is mainly intended to make possible retrieval of Afan Oromo text documents by applying techniques of modern information retrieval system. Information retrieval is a mechanism that enables finding relevant information material of unstructured nature that satisfies information need of user from large collection.

Afaan Oromo text retrieval developed in this study has indexing and searching parts. Vector Space Model of information retrieval system was used to guide searching for relevant document from Oromiffa text corpus. The model is selected since Vector space model is the widely used classic model of information retrieval system. The index file structure used is inverted index file structure.

For this study text document corpus is prepared by the researcher encompassing different news article and experiment is made by using 9(nine) different user information need queries. Various techniques of text pre-processing including tokenization, normalization, stop word removal and stemming are used for both document indexing and query text.

The experiment shows that the performance is on the average 0.575(57.5%) precision and 0.6264(62.64%) recall. The challenging tasks in the study are handling synonymy and polysemy, inability of the stemmer algorithm to all word variants, and ambiguity of words in the language.

The performance 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.

Table Contents

List of Tables

List of Figures

List of Algorithms

List of Codes

CHAPTER ONE

INTRODUCTION

1.1 Background

Information Retrieval (IR) is one of the major branches of Information Science discipline [1]. The trend in information storage and retrieval can be traced back to 2000 BC when people of Sumerians chose special place to store clay tablets with cuneiform inscription [2]. After they understand their work is efficient on use of information, they developed special categorization system that identifies every tablets and its content.

One of the major evolutions in Information Retrieval is invention of print machine in 1450 A.D [3]. A German goldsmith Johannes Gutenberg invented the first movable printer, thousand years later after Chinese invented paper which provides means for disseminating and storing knowledge. Gutenberg's aim was allowing direct access to mass information that was contained in the Bible and other scholarly works. The invention of print machine ignited the Reformation and Renaissance.

With the introduction of print machine, print materials are flourished out enormously than ever before. Accessing library materials was a big problem in those days because of increase in library holdings. In order to simplify accessing library collection classification scheme developed include Dewey Decimal Classification (DDC) created by Melvin Dewey in 1876[1], Library of Congress Classification Scheme (LC)[2], and Universal Decimal Classification (UDC)[3].

The development of modern IR is highly related to World War II (WWII) in 1945 and cold war after end of the actual war. Because of the difficulty in storing and retrieving large numbers of scientific papers publications during post war, devising new mechanism become mandatory task. In fact most the papers are published for the interest of US military and it should be accessible to them easily. That is why Venavar Bush, Head of National Science Foundation (NSF) defined the problem as 'Information explosion'-- mass production of information [4]. The challenge of information explosion is finding document relevant to information need users. Bush designed the machine called MEMEX in 1945 to solve the

[1] DDC is maintained by OCLC(Online Computer Library Center), available at http://www.oclc.org
[2] USA's National Library , available at http://www.loc.gov/index.html
[3] UDC Consortium , available at http://www.udcc.org/

problem of information explosion. MEMEX is "*a hypothetical, desk size information work-station, contained a massive amount of microfilm, all libraries' worth; including purchased material as well as personal documents that would be scanned or penned in*" [5:106]. Documents called using their code by rapid selector. The concept of MEMEX is fully realized in modern hypertext of World Wide Web. This is considered as the starting point for modern IR. These days information explosion is even increasing. Facts show that print materials are being replaced by electronic one and accessing a single document of interest is being difficult, because it is unthinkable to check manually each and every possible document available to us [4].

Hans Peter Luhn research engineer at International Business Machine Corporation (IBM) started mechanized punch card based system for searching chemical compounds in 1947. The term Information Retrieval is claimed as it was coined by Calvin Mooers in 1950 for the first time. It is on the "International Conference on Scientific Information" Washington DC, 1958, that IR system is identified as a solution to retrieval problems. Sooner in 1960 Melvin Earl (Bill) Maron and J. L. Kuhns published "On relevance, probabilistic indexing, and information retrieval" and in 1962 Cyril W. Cleverdon developed a model for IR system evaluation. In 1963, Joseph Becker and Robert Hayes publish "Information Storage and Retrieval: tools, elements and theories". The concept of hypertext is promoted by Theodor Nelson early 1970's. Tim Berniers Lee presented the first proposal on World Wide Web at CERN (European Council for Nuclear Research) in 1989. Late 1990's implementation of web search engines was getting more emphasis [3][5].

Information retrieval is defined as finding documents of an unstructured nature that satisfies information need of users from within large collection [6]. An information retrieval system is system that stores and manages information on documents and also enables users finding the information they need. It returns documents that contain answer to users question rather than explicit answer to their information need. Most of the time retrieved documents satisfy users' information needs. Documents which satisfy users' information needs called relevant documents, whereas documents which are not satisfying users' information need are irrelevant documents. In fact there is no perfect information retrieval system which retrieves all relevant documents and no irrelevant document [7].

Information Retrieval has two main subsystems [6], Indexing and Searching. Indexing is an offline process 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. Searching is the process of relating index terms to query terms and return relevant hits to users query. Both indexing and searching are interrelated and dependent on each other for enhancing effectiveness and efficiency [6].

The focus of quality of Information Retrieval design is in evaluating both retrieval effectiveness and efficiency [6]. Efficiency is about optimizing computing resource such as the needed storage space and time complexity, while effectiveness concerned with relevancy of document retrieved that satisfies users' information need.

Information Retrieval is fast growing subfield of Information Science that mainly deals with searching and access to relevant information from unstructured corpus of documents [6]. The survey made in 2004 by Pew Internet reveals that 92% of Internet users use search engines to find information they need. The size of World Wide Web is increasing exponentially. The price of major scientific innovations and computer hardware is dramatically declining. In spite of the entire above, there are many powerful commercial search engines which are able to provide high quality results for millions of users within seconds [6].

Nowadays amazing progress has been seen on the development of information retrieval systems. The retrieval system is moving from classical retrieval models to toward intelligent techniques. The commercial search engines are well organized. Performance of commercial search engines in both efficiency and effectiveness is very high these days. For instance Google[4], Bing[5], Yahoo[6], and Amazon[7] are search engines with high reputations. Additionally there are many desktop retrieval systems including Windows Search, Google Desktop Search, Yahoo! Desktop Search, Locate32, Search Everything and much more. These searches either the desktop version or the web version works well for English and some other international languages. Google has interface in Afaan Oromo[8] but the system works not well.

[4] General purpose search engine, www.google.com
[5] General purpose search engine, www.bing.com
[6] General purpose search engine, www.yahoo.com
[7] Online store and market, www.amazon.com
[8] Afaan Oromo ('Oromo Language', or 'Oromic' in English) and also known as 'Oromo', 'Oromiffa', and 'Afan Orma'

There are different challenges in implementing IR system. Information retrieval is language dependent process which needs integrating knowledge of information retrieval techniques and natural language. Most of IR techniques are developed for English language and it is always difficult task applying it for other languages. The other thing is tradeoff between efficiency and effectiveness in terms of IR system performance. IR system should be both effective and efficient but always increasing decreases the other. So coming up with efficient effective system needs tough task. Additionally the evaluation performance IR system is also challenging task, because performance of IR is evaluated in relative to relevance of retrieved document toward users' query and efficiency. But it is really difficult to identify what is relevant from the irrelevant one, because human information needing behavior is fluctuating.

1.2 Statement of the Problem

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 [9] 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.

Afaan Oromo uses Latin based script called "Qubee" and it has 26 basic characters. It is the official language of Oromia regional state of Ethiopia and also academic language for primary school of the region. Oromo language, literature and folklore delivered as a field of study in many universities located in Ethiopia and other countries [10]. Nowadays journal, magazines, newspapers, news, online education, books, entertainment Medias, videos, pictures, are available in electronic format both on the Internet and on offline sources. There is huge amount of information being released with this language, since it is the language of education and research, language of administration and political welfares, language of ritual activities and social interaction [11].

There are a number of Medias that uses Afaan Oromo as 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 [11].

Development of language is highly related with development of technology. The fact that initiated this study is also enabling development of Afaan Oromo to grow with current information technology support. IR is not being optional technology, it is something that is

very important to everybody and mandatory to use. In this Information Age, information is highly needed than anything else. But finding this important information needs system support [12].

In Ethiopia there is an attempt to develop Information Retrieval system for Amharic Language [13][14][15], the official language of Ethiopia. The work done for Amharic language includes development of retrieval algorithms for Amharic documents which are mainly written in Ethiopic (Ge'ez script) characters. Additionally there is also attractive attempt to develop search engine for the same language [16].

There is works done for Afaan Oromo specifically on Cross lingual Information Retrieval system. Prior to these works there was no other IR system for the Afaan Oromo. Attempt made by the study is to enable retrieval of document of other languages specifically English in Afaan Oromo query. But this doesn't solve need of users all in all. Users whose looks for Afaan Oromo document with Afaan Oromo query may not find suitable environment to find information of their need. The CLIR algorithm used follows either of machine translation or parallel corpuses. From perspective of IR principles it is important to have system that works for Afaan Oromo and then it is better extending it to CLIR. Implementation of this work helps users of Afaan Oromo to find information of their need simply without much difficulty.

Hence the aim of this study is to develop a prototype for Afaan Oromo text retrieval system that organize document corpus using indexing and search relevant ones as per users query based on vector space model.

To this end, this study tries to answer the following research questions.

- What are the unique features of word formations in the Afaan Oromo writing system?
- What are the suitable text operations and indexing scheme to apply for organizing document corpus?
- To what extent vector space model enables to design effective Afaan Oromo text retrieval?

1.3 Objective of the Study

The present study is mainly intended to make possible retrieval of Afaan Oromo documents from document corpus by using modern information retrieval techniques.

1.3.1 General Objective

The main objective of this study is to come up with an Information Retrieval system that can enable to search for relevant Afaan Oromo text corpus. Application of techniques of modern information retrieval techniques enables to solve problems related with accessing information that satisfies information needs of Afaan Oromo users.

1.3.2 Specific Objectives

In order to accomplish the above stated 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.
- To understand basics of Afaan Oromo language and perform text operation such as, tokenization, stop words removal, stemming and normalization.
- To design an architecture for implementing Afaan Oromo Information Retrieval System.
- To develop a prototype IR system that searches referent documents from unstructured corpus.
- To evaluate the performance of the proposed prototype and recommend future research direction.

1.4 Scope and Limitation of the Study

The focus of this study is on designing an information retrieval system that effectively searches with in Afaan Oromo text corpus. the work mainly implements an indexer and searcher from corpus of Afaan Oromo textual documents. The study involves both indexing and searching part. 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 are applied. Index terms are organized using inverted index file and searching for documents satisfying query terms are guided by vector space model.

Additionally the study does not consider relevance feedback, artificial intelligence techniques to predict users' information need and profiling information seeking behavior of users.

As a result of time factor, limited corpus was used for evaluating the performance of the IR system developed in the study. This is because it takes more time to prepare relevance judgment for corpus with many documents.

1.5 Methodology

Methodology is a way to systematically solve the research problem [17]. This research was conducted in order to figure out challenges of implementing Afaan Oromo retrieval system. Towards achieving the main objective the following step by step procedures are followed.

1.5.1 Literature Review

To have conceptual understanding and identify the gap that is not covered by previous studies different materials, including journal articles, conference papers, books, and the Internet have been reviewed. In this study the review is mainly concerned works that have direct relation with the topic and the objective of the study. These include previous works done on local information retrieval system giving more attention Afaan Oromo IR.

1.5.2 Corpus/Data Set Preparation

The corpus size is 100 text documents written in Afaan Oromo language and with Latin alphabets which is called 'Qubee'. The corpus is built from the official website of Oromiya National Regional State[9], Voice of America Radio Afaan Oromo language[10], Gumii Waaqeffattoota Addunyaa (GWA) Portal[11], International Bible Society official website[12], Oromiya Radio and Television Organization (ORTO)[13] news and other Internet based sources.

Data set used is mainly, news articles, others resources from Holy Bible books and the Internet. None of the selected documents are domain specific, rather it covers different aspect of life like sport, culture, socio-economic, political, religious, education and development areas. Heterogeneity of the data set helps evaluation of the system more generic.

[9] http:// www.oromiyaa.com
[10] http://www.voanews.com/oromoo
[11] http://waaqeffannaa.org
[12] http://www.biblica.org
[13] http:// www.orto.gov.et

1.5.3 Development tools

The development area used is Windows environment and the programming language used is Python 2.7.3. Python is dynamic programming language that is used in a wide variety of application domains. It is simple, strong, involves natural expression of procedural code, modular, dynamic data types, and embeddable with in applications as a scripting interface [18].

1.5.4 Evaluation Techniques

The study involves developing the designed system and evaluating its performance. To this end, corpus is prepared, queries are constructed and relevance judgment is made for evaluating effectiveness of the work to measure the effectiveness of the IR system precision and recall are used. Precision is fraction of retrieved documents that are relevant, and recall is fraction of relevant document that retrieved [3]. In this work the interpolated precision value will be used to draw precision-recall curve in order to evaluate retrieval effectiveness of the system. To measure performance of the system for multiple queries an average precision is calculated.

1.6 Significance of the study

Development of language is directly related with adaptation of ongoing technology and making it suitable to the local language. Speakers' of language are no more interested in speaking language of technology, rather they need technology speak their own language. If the language manage to grow with technology speakers of the specific language will be benefited from the development in many ways. That is why it is important localizing works already done in developed languages like English. Generally the study has the following significance:

- Opens way for others researchers to focus on the area and work on it, it can be used as stepping stone for further work in this specific area.
- Enables Afaan Oromo speakers retrieving text documents in Afaan Oromo efficiently and effectively
- Improves the development of the language
- Since the study is master theses it has also learning out comes for the researcher. It helps the researcher to investigate problems and solving it in scientific manner.

1.7 Organization of the Thesis

The study is organized in the following way. The chapter one is introductory part. In this part basic concept about IR system, what initiated the study-- the statement of the problem, objective of the study, scope of the study, research methodologies and study's significance is discussed.

The rest of the thesis is organized in the following way. Chapter Two is literature review and it involves two main topics, related works and conceptual review. Conceptual review is review on Information retrieval system and related topics, and Afaan Oromo's related concepts. Related work involves work done so far on the research topic.

Major technique and methods used in this study are discussed in chapter three. In this section techniques used for indexing and searching are discussed. Additionally term weighting technique (tf*idf), retrieval model (VSM), and the similarity measurement (cosine similarity) discussed in this part of the paper.

The fourth chapter of the work is Experimentation and result of the study. In this part dataset selections and preparations, implementations of the proposed work, experimentations, findings of the study, issues in in implementations are discussed in detail.

Finally in the chapter five major findings including faced challenges are written as a conclusion and works identified as future work and needs to get attention of other researchers are listed in recommendation section.

CHAPTER TWO

LITERATURE REVIEW

The information retrieval is very vast area of study, with the main aim of searching for relevant documents from large corpus that satisfies information need of users. Since the IR research involves language dependent process, in this study we attempt to design an IR system Afaan Oromo language.

2.1 Afaan Oromo

Afaan Oromo is Cushitic language which is family of Afro Asiatic languages. It is spoken by more 30 Million peoples and most of native speakers are people living in Ethiopia, Kenya, Somalia and Egypt. It is third largest language in Africa following Kiswahili and Hausa; 4th largest language, if Arabic is counted as Africa language [11][19].

The exact time when the Latin alphabet started being used for Afaan Oromo writing was not well known, but on November 3, 1991 it adopted as official alphabet of Afaan Oromo on. Now it is language of public media, education, social issues, religion, political affairs, and technology.

2.1.1 Dialects and Varieties

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 [20].

2.1.2 Alphabets and Sounds

Oromo is phonetic language that is spoken in 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.

Until the 1970s Afaan Oromo was written with either the Ge'ez script or the Latin alphabet. From 1974- 1991 writing of Oromo in any script was forbidden. Since 1991 Latin alphabet is used as official alphabet of Oromo Language [10].

Afaan Oromo uses Latin character (Roman alphabet) but with some modifications on sound of consonant and vowels. It has 28 letters called 'qubee'. However, later on a new letter'Z' was included in the alphabet as there are words which require the letter. For example: Ziqaya (gold), Zeeytuuna (guava), Azoole (river in Arsii), Zeekkara (Opera), Zalmaaxaya (mess), Waziiza (fire place or fire work) and Zawii (insanity) are Afaan oromo words written using 'Z'. Additionally 'P' and 'V' are also added. 'P' and 'V' letters are not Afaan Oromo letters because there is no Oromo word written by use of either of them. But they are included by considering the fact of handling borrowed terms from other languages like English. For example: 'Police', 'Piano', 'Television', 'video', and etc. To sum up there are 31 letters of Afaan Oromo including 'Z', 'P', and 'V' [21].

All Afaan Oromo phonetic sound are available in Appendix I. Here in Table 2.1 alphabets are listed in both Upper case and Lowercase. Alphabets sound is also included with how it pronounced in English words [10]. Audio based sound is also available online[14].

Alphabets	***Sounds***	**Alphabets**	***Sounds***	**Alphabets**	***Sounds***	**Alphabets**	***Sounds***	**Alphabets**	***Sounds***	**Alphab**	***Sounds***
A a	*[aa]* like **ask**	**B b**	*[baa]* like **bi**rd	**C c**	*[Caa]* like **ca**t	**D d**	*[daa]* like **da**m	**E e**	*[ee]* like **a**te	**F f**	*[ef]* like **fu**ngi
G g	*[gaa]* like **gu**n	**H h**	*[haa]* like **ha**t	**I i**	*[ie]* like **I**ndia	**J j**	*[jaa]* like **Ju**st	**K k**	*[kaa]* like **Ca**st	**L l**	*[la]* like **li**fe
M m	*[ma]* like **ma**n	**N n**	*[naa]* like **na**sty	**O o**	*[oo]* like **o**ld	**P p**	*[pee]* like **pa**st	**Q q**	*[quu]* like **quit**	**R r**	*[ra]* like **ra**t
S s	*[saa]* like **sa**lad	**T t**	*[taa]* like to**ta**l	**U u**	*[uu]* like **u**rge	**V v**	*[vau]* like **va**ry	**W w**	*[wee]* like **wa**nt	**X x**	*[taa]* like ___
Y y	*[y]* like **you**th	**Z z**	*[Zay]* like **Tha**t	**CH ch**	*[chaa]* like **cha**t	**DH dh**	*[dhaa]* like ___	**SH sh**	*[shaa]* like **sh**y	**NY ny**	*[nyaa]* like _____
PH ph	*[phaa]* like __										

Table 2. 8 Upper Case, lower case and their sounds

[14] http://en.wikibooks.org/wiki/Afaan_Oromo/Alphabet

A. Vowels-Dubifttu

Afaan Oromo vowels are similar to that of English but sound differently. There are five vowels **a, e, o, u** and **i**. All vowels pronounced in the similar way throughout every Afaan Oromo literature. These vowels pronounced in sharp and clear fashion which means, each and every word is pronounced strongly.

A: *A*r'bb*a*, F*a*rdd*a*, H*aa*dh*a*

E: Gannal*e*, Waab*ee*, Nool*e*, Roobal*e*, coll*ee*

I: Ars*ii*, laal*i*, Raf*i*, Lakk*i*, S*i*rbb*i*

O: *O*rom*o*, Cilaal*o*, Har*o*, caancc*o*, Danbid*oo*ll*o*

U: *U*lfaadh*u*, G*u*dadh*u*, d*u*badh*uu*, g*uu*g*u*, It*uu*

Vowels			
	Front	**Central**	**Back**
close	i /ɪ/, ii /iː/		u /ʊ/, uu /uː/
Mid	e /ɛ/, ee /eː/		o /ɔ/, oo /oː/
Open		a /ʌ/	aa /ɑː/

Table 2. 9 Dubbiistoota (Vowels)

B. Consonants –Sagaleewwan (dubbifamtoota)

Most Afaan Oromo constants do not differ greatly from Italian, but there are some exceptions and few special combinations.

i. The consonant "g" has a hard sound. Gaari, gadi bayi, gargaari.

ii. The combinations NY and DH have a hard sound. e.g Nyaadhu, Dhugi.

Consonants						
		Bilabial/ Labiodental	**Alveolar/ Retroflex**	**Palato-alveolar/ Palatal**	**Velar**	**Glottal**
Stops and affricates	Voiceless	(p)	T	ch /tʃ/	k	' /ʔ/
	Voiced	b	D	j /dʒ/	g	
	Ejective	ph /p'/	x /t'/	c /tʃ'/	q /k'/	
	Implosive		dh /ɗ/			
Fricatives	Voiceless	f	S	sh /ʃ/		h
	Voiced	(v)	(z)			
Nasals		m	N	ny /ɲ/		
Approximants		w	L	y /j/		
Rhotic			R			

Table 2. 10 Dubbifamtoota (Consonants)

2.1.3 Grammar

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

Gender

Oromo has two grammatical genders, masculine and feminine in similar way to other Afro-Asiatic language. There is no neutral gender as in other language like English [21]

Small number of nouns ends with *-eessa* (m.) and *-eettii* (f.), as do adjectives when they are used as nouns: *obboleessa* 'brother', *obboleettii* 'sister', *dureessa* 'the rich one (male.)', *hiyyeettii* 'the poor one (female). Grammatical gender normally agrees with biological gender for people and animals; thus nouns such as *abbaa* 'father', *ilma* 'son', and *sangaa* 'ox' are masculine, while nouns such as *haadha* 'mother' and *intala* 'girl, daughter' are feminine. However, most names for animals do not specify biological gender [21].

Names of astronomical bodies are feminine: *aduu* 'sun', *urjii* 'star'. The gender of other inanimate nouns varies somewhat among dialects.

Number

There are singular and plural numbers in Afaan Oromo. But nouns that refer to multiple entities are not obligatorily plural. Some singular noun may refer multiple entities: “*nama*” “man” or “people”, “*nama shan*” “five men” or “five people” [21].

When it is important to make the plurality of a referent clear, the plural form of a noun is used. Noun plurals are formed through the addition of suffixes. The most common plural suffix is *-oota*; a final vowel is dropped before the suffix, and in the western dialects, the suffix becomes -ota following a syllable with a long vowel: *mana* 'house', *manoota* 'houses', *hiriyaa* 'friend', *hiriyoota* 'friends', *barsiisaa* 'teacher', *barsiiso(o)ta* 'teachers'. Among the other common plural suffixes are *-(w)wan*, *-een*, and *-(a)an*; the latter two may cause a preceding consonant to be doubled: *waggaa* 'year', *waggaawwan* 'years', *laga* 'river', *laggeen* 'rivers', *ilma* 'son', *ilmaan* 'sons'[21].

Definiteness

There is no indefinite article (such as *a, an, some)* in Afaan Oromo like they exist in English. The definiteness article *‘the’* in English is *(t)icha* for masculine nouns (the *ch* is geminated

though this is not normally indicated in writing) and *-(t)ittii* for feminine nouns in Afaan Oromo. Vowel endings of nouns are dropped before these suffixes: *karaa* 'road', *karicha* 'the road', *nama* 'man', *namicha*/*namticha* 'the man', *haroo* 'lake', *harittii* 'the lake'. Note that for animate nouns that can take either gender, the definite suffix may indicate the intended gender: *qaalluu* 'priest', *qaallicha* 'the priest (male)', *qallittii* 'the priest (female)'. The definite suffixes appear to be used less often than *the* in English, and they seem not to co-occur with the plural suffixes [21].

Personal pronoun

Oromo pronouns include personal pronouns (refer to the persons speaking, the persons spoken to, or the persons or things spoken about), indefinite pronouns, relative pronouns (connect parts of sentences) and reciprocal or reflexive pronouns (in which the object of a verb is being acted on by verb's subject).

English	**Base**	**Subject**	**Dative**	**Instrumental**	**Locative**	**Ablative**	**Possessive adjectives**
I	*ana, na*	*ani, an*	*naa, naaf, natti*	*naan*	*natti*	*narraa*	*koo, kiyya* [*too, tiyya* (f.)]
you (sg.)	*si*	*ati*	*sii, siif, sitti*	*siin*	*sitti*	*sirraa*	*kee* [*tee* (f.)]
he	*isa*	*inni*	*isaa, isaa(tii)f, isatti*	*isaatiin*	*isatti*	*isarraa*	*(i)saa*
she	*isii, ishii, isee, ishee*	*isiin*, etc.	*ishii, ishiif, ishiitti*, etc.	*ishiin*, etc.	*ishiitti*, etc.	*ishiirraa*, etc.	*(i)sii, (i)shii*
we	*nu*	*nuti, nu'i, nuy, nu*	*nuu, nuuf, nutti*	*nuun*	*nutti*	*nurraa*	*keenna, keenya* [*teenna, teenya* (f.)]
you (pl.)	*isin*	*isini*	*isinii, isiniif, isinitti*	*isiniin*	*isinitti*	*isinirraa*	*keessan(i)* [*teessan(i)* (f.)]
they	*isaan*	*isaani*	*isaanii, isaaniif, isaanitti*	*isaaniitiin*	*isaanitti*	*isaanirraa*	*(i)saani*

Table 2. 11 Afaan Oromo personal pronouns

Adjectives

Adjectives are very important in Afaan Oromo because its structure is used in every day conversation. Oromo Adjectives are words that describe or modify another person or thing in the sentence [21].

Colors--bifoota	**Sizes--hamma guddina**	**shapes -- boca**	**Tastes--**	**Qualities--**
English-*Afaan Oromo*	English-*Afaan Oromo*	English-*Afaan Oromo*	English-*Afaan Oromo*	English-*Afaan Oromo*
Black-- *gurraacha*	big --*guddaa*	circular-*geengoo*	bitter --*hadhaawaa*	bad --*hamaa*
Blue-- *baluu*	deep--	straight --*sirrii*	fresh --*asheeta*	clean --*qulqulluu*
Brown-*bifa bunaa*	*gadifagoo*	square--*golarfee*	salty --*sogidaawaa*	dark-- *dukkanaawaa*
Gray -- *daalacha*	long-	triangular-*golsadee*	sour --*oganaawaa*	difficult --*ulfaataa*
Green-- *magariisa*	*lafarradheerat*	E.g	spicy--*kaurgoo*	dirty --*xuraawaa*
Red --*diimaa*	*a*	The circular	*baayyatu*	dry-- *qooraa*
White --*adii*	narrow --	house→ *manicha*	sweet --*mi'aawaa*	easy --*salphaa*
Eg.	*dhiphaa*	*geengoo*	E.g	empty --*duwwaa*
A green tree→ muka magariisa	short --		a salty food→	expensive--
	gabaabaa		*nyaata hadhaa'aa*	*mi'aa/qaalii*
	small --*xinnaa*			E.g
	tall --*dheeraa*			a very nice friend→
	thick --*yabbuu*			*jaala baayyee*

Table 2. 12 Afaan Oromo Adjectives

Adverbs (Ibsa Xumuraa)

Oromo adverbs are part of speech. Generally they're words that modify any part of language other than a noun. Adverbs can modify verbs, adjectives (including numbers), clauses, sentences and other adverbs[21].

The four categories of adverbs in Afaan Oromo: *Adverb of time, adverbs of place, adverbs of manner and adverbs of frequency.* Lists of adverbs are given in table 2.5. The lists on left side are in English while those on the right side equivalent meaning of the terms in Afaan Oromo.

Adverbs of time			
English	**Afaan Oromo**	**English**	**Afaan Oromo**
Yesterday	*kaleessa*	**Adverbs of manner**	
today	*harr'a*	very	*baayyee*
tomorrow	*bor*	quite	*baayyee*
Now	*amma*	really	*dhugumaan*
then	*gaafas*	fast	*dafee*
later	*eger*	well	*gaarii*
tonight	*edans*	hard	*cimaa*
right now	amma *isa ammaa*	quickly	*dafee*
last night	*eda*	slowly	*suuta*
this morning	*ganam kana*	carefully	*qalbiidhan*
next week	*torban dhufu*	absolutely	*matuma*
recently	*dhiyeenya kana*	together	*walii wajjin*
soon	*dhiyootti*	alone	*qophaa*
immediately	*hatattamaan*	**Adverbs of frequency**	
Adverbs of place		always	*yeroo hunda*
here	*as*	sometimes	*gaaffii gaaf*
there	*achi*	occasionally	*gaaffii gaaf*
over there	*gara sana*	seldom	*darbee darbee*
everywhere	*iddoo hunda*	rarely	*darbee darbee*
nowhere	*eessayyu*	never	*yoomiyyuu*
home	*mana*		
away	*fagoo*		
out	*ala*		

Table 2. 13 Adverbs in Afaan Oromo

Prepositions

Prepositions in Afaan Oromo, links nouns, pronouns and phrases to other words in a sentence. The word or phrase that the preposition introduces is called the object of the preposition[21]. Here are list of some prepositions:

English Prepositions	Oromo Prepositions
about	*waa'ee*
above	*gubbaa / gararraa*
across	*gama*
after	*booddee / booda*
against	*faallaa*
among	*jara giddu*
around	*naannoo*
as	*akka*
at	*itti*
before	*dura*
behind	*dudduuba / dugda duuba*
below	*jala / gajjallaa*
beneath	*gajjallaa*
beside	*bira*
between	*gidduu*
beyond	*garas*
but	*garuu*
by	*..dhaan*
despite	*ta'uyyuu*
down	*lafa*
during	*utuu*
except	*malee*
for	*f*
from	*irraa*
in	*keessa*
inside	*keessa*
into	*keessatti*
near	*bira*
next	*ittaanee*
of	*kan*
on	*irra*
opposite	*fuullee / faallaa*
out	*ala*
outside	*alla*
over	*irraan*
per	*..tti*
plus	*fi*
round	*naannoo*

English Prepositions	Oromo Prepositions
since	*ergii*
than	*mannaa*
through	*gidduu*
till	*hamma*
to	*tti*
toward	*garas*
under	*jala / gajjallaa*
unlike	*faallaa*
until	*hamma*
up	*gubbaa*
via	*karaa*
with	*wajjin*
within	*keessatti*
without	*malee*
two words	*jechoota lama*
according to	*akka kanaatti*
because of	*kanaaf*
close to	*bira*
due to	*kanaaf*
except for	*kana malee*
far from	*irraa siqee / iraa fagaatee*
inside of	*keessa isaa*
instead of	*kanaa mannaa*
near to	*itti aanee*
next to	*itti aanee*
outside of	*kanaa alatti*
prior to	*kanaan dura*
three words	*jechoota sadii*
as far as	*hamma*
as well as	*fi*
in addition to	*dabalatees*
in front of	*fullee isaa*
in spite of	*ha ta'uyyu malee*
on behalf of	*maqaa …*
on top of	*kana irraan*
demonstrative prepositions	*agarsiisoo*
this	*kana / tana*
that	*sana*
these	*warra kana / jara kana*

Table 2. 14 Afaan Oromo Prepositions

Oromo prepositions are used in similar way it us fed in English and written separately from root word. So it is easy to remove from content bearing terms easily, as stop words. In some cases propositions are connected with root words.

Negation

Oromo *negation* is the process that turns an affirmative statement (I am happy) into its opposite denial (I am not happy). In most cases negation is formed by adding prefix '*hin'* to the verb[21].

Examples:

I don't speak –*hin dubbadhu*	he doesn't write-- *inni hinbarreessu*
I don't write-- *hin-oofu*	we don't speak—*hin dubbannu*
he doesn't speak—*hin dubbatu*	we don't write—*hin barreessinu*

2.2 Overview of IR system

IR is concerned with searching of documents for information from document corpus and the World Wide Web. IR searches both structured and unstructured information.

IR system includes various process and techniques. The whole IR system includes two main subsystem: Indexing and searching[6]. Indexing is the process of preparing index terms which are either content bearing or free text extracted from documents corpus. Searching is process matching users information via query to documents in collection via index terms. These searching and indexing processes is guided by model, Information Retrieval Model. There are various IR models including VSM, Probabilistic and Boolean to list a few. Additionally it is important to measure performance of IR system. The performance evaluation techniques help us to know accuracy of the IR system. Figure 2.1 shows IR system component and their interrelation.

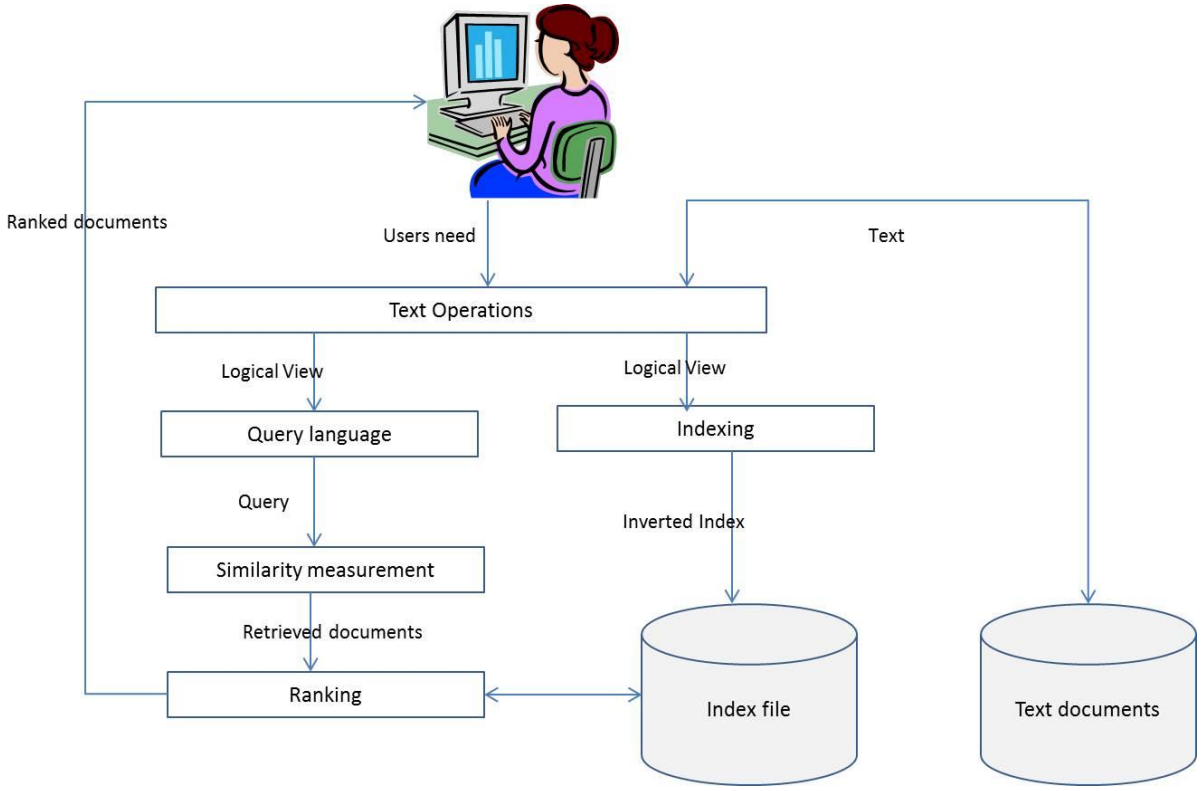


Figure 2. 1 General Architecture of Information Retrieval System [22]

2.3 Indexing

Indexing is an offline process of extracting index terms from document collection and organize them using indexing structure to speed up searching [6]. It is language dependent process which varies from language to language [23].

Three of the most commonly used file structures for information retrieval can be classified as lexicographical indices (indices that are sorted), clustered file structures, and indices based on hashing[2]. One type of lexicographical index is the inverted file, and the other one is lexicographical index, the suffix tree (PAT tree). Both are reviewed in this study because of its higher interconnection to study.

Suffix tree

Suffix tree [6](also called PAT tree) is a data structure that presents the suffix of a given in a way that allows for a particularly fast implementation other operation like search. The suffix tree for a string S is a tree whose edges are labeled with strings, such that each suffix of Scorresponds to exactly one path from the tree's root to a leaf.

Constructing such a tree for the string Stakes time and space linear in the length of S. Once constructed, several operations can be performed quickly, for instance locating a substring in S, locating a substring if a certain number of mistakes are allowed, locating matches for a regular expression pattern etc. Suffix trees also provided one of the first linear-time solutions for the longest common substring problem. These speedups come at a cost: storing a string's suffix tree typically requires significantly more space than storing the string itself [6].

Inverted index

An **inverted index** is an index data structure storing a mapping from content, such as words or numbers, to its locations in a document or a set of documents. The purpose of an inverted index is to allow fast full text searches, at a cost of increased processing when a document is added to the corpus. The inverted file may be the database file itself, rather than its index. It is the most popular data structure used in document retrieval systems, used on a large scale for example in search engines [6].

There are two different kind of inverted files [24]. One is record level inverted index (or inverted file index or just inverted file): contains a list of references to documents for each word. The other is a word level inverted index (or full inverted index or inverted list) additionally contains the positions of each word within a document. Word level inverted index enables more functionality (like phrase searches), but needs more time and space to be created [24].

The process of construction of inverted index follows the four critical steps [23]: primarily, collecting document to be indexed, secondly, tokenization of the text, turning each document in to a list of tokens, thirdly, Linguistic preprocessing, producing a list of normalized and stemmed tokens, which are the index terms, and finally index the documents that each term occurs in by creating an inverted index, consisting of a directory and postings.

Tokenization: it is the process of chopping on white spaces and throwing away punctuation characters. For example, if the original document is "*Oromiyaan, magaalaa fi baadiyyaa qabdi*" the tokens will be '*Oromiyaan*', '*magaalaa*','fi' *baadiyyaa*', '*qabdi'*. So we can define *token* as an instance of a sequence of characters. Each such token is now a candidate for an index entry, after further processing. One of challenges related to

tokenization is--differentiating single word from compound word, for instance *Hewlett-Packard* Versus *Hewlett and Packard, AddisAbaba* Versus *Addis Ababa.* The other challenge is identifying numerical values like dates, phone numbers, and IP addresses. Additionally Chinese and Japanese has no space between words, which makes difficult space and punctuation mark based tokenization. Arabic and Hebrew are basically written right to left, but with certain items like numbers written left to right; the challenge here is it is not possible to use the same algorithm used for Latin based language for such languages too. That is why tokenization is called natural language dependent technique [25].

Preprocessing includes case folding/normalization, stop word removal and stemming.

Case Folding/ Normalization: is the process of handling problem related with variation cases (UPPER CASE, or lower lase or Mixed Cases). So the good way to handle this problem is converting the whole document in to similar case. Often best to lowercase everything, since most of the time users use lowercase regardless of 'correct' capitalization. Here are some examples: Republican vs. republican, *John* vs. *john vs. JOHN*, and etc. In some languages like Amharic which does not have a distinction between upper and lower case this might not be a big deal. But it is very important for languages using Latin characters [26][27].

Stop-words: Stop-words are most frequent terms which are common to every **document**, and have no discriminating power one document from the other. So these terms should not be considered in *indexing* process. According to *Zipf's law* [23] few terms occur frequently, a medium number of terms occurs with medium frequency and many terms with very low frequency. This shows that writers use limited vocabulary throughout the whole document, in which even fewer terms used more frequently than others. Further Luhn defined word significance across the document in 1958 [23]. Luhn suggested that both extremely common and extremely uncommon terms are not very useful for indexing. So we should upper and lower cut-off points. Upper cutoff enables to remove the most frequent terms whereas, a lower cut-off controls less frequent terms which believed to be non-content bearing terms [1].

Removing **stop-word** from English document saves disk space 25%-30%. But it is not good if the system handle phrase query, song names, etc. For example *As we may* think, *to be or not to be* [6].

Stemming: stemming is process used in most search engines and information retrieval systems. It is core natural language processing technique for efficient and effective IR system. Generally stemming transforms inflated words in to their most basic form. There are different stemming algorithms but the most common one is that of Porter, called 'Porter Stemmer'. Even if stemming is very similar to lemmatization in most of indexing process stemming is used.

Stemming is language dependent process in similar way to other natural language processing techniques. It is often removing inflectional and derivational morphology. E.g. automate, automatic, automation → *automat*. Stemming has both advantage and disadvantage. The advantage is it helps us to handle problems related to inflectional and derivational morphology. That makes words with similar stem/root word to retrieve together. This increases effectiveness IR system. Stemming has disadvantage sometimes: some terms might be over stemmed, this changes meaning of the terms in the document; different terms might be reduced to the same stem, which still enforce the system as to retrieve non relevant documents [28].

The inverted index data structure is core component of a typical search engine indexing algorithm. It improves quality of search engines by increasing speed of searching to find documents with term occurrence. With the inverted index created, the query can be resolved by jumping to the word id (via random access) in the inverted index.

2.4 Document Representation and Term Weighting

There are different ways of document representation. Document representation helps us to give different weight for different terms with respect to given query. It helps judge as if document is either relevant or not with respect to users query. In this work ***tf*idf*** term weighting is used for determining relevance of the terms[29].

There are different mechanisms of assigning weight to terms.

i. Binary Weights

- Only the presence (1) or absence (0) of a term is included in the vector

ii. Raw term frequency

- Only the presence (1) or absence (0) of a term is included in the vector

*iii. tf * idf*

- Term Frequency (tf) *Inverse Document Frequency (idf)

Even if there are three different ways of calculating term weight in this study *tf *idf* weighting technique has been used. The reason why *tf *idf* weight is selected is because it is normalized weighting technique and it is standard way of calculating weight.

The term frequency*inverse document frequency also called **tf*idf**, is a well know method to evaluate how important is a word in a document in Information Retrieval and text mining. It is statistical method which reflects how important a word/term is to a document in collection/corpus. ***tf*idf*** is also a very interesting way to convert the textual representation of information into a Vector Space Model (VSM). The value of ***tf*idf*** increases with proportion to the number of times a word appears in the document and decreases as the term exist frequently in the document corpus. The more common terms which exist in almost all documents has lower score of *tf*idf*, whereas terms exist frequently in single but not in others have higher score [24][29].

Generally speaking ***tf*idf*** is used for term weighting. This weighting technique is used often by search engines and information retrieval systems. It can be used for filtering stop-words and also have application in text summarization and classification [29].

For example if the user is *"the brown cow"* it is only documents that contains terms *"the", "brown",* and *"cow",* which should be considered as relevant. So other documents which are not having any of these three terms are excluded from retrieval. In order to distinguish importance level of document frequency of each terms with (*term frequency*) in each documents counted [25].

In the above example the term *'the'* is common word and obviously the most frequent term, if it is considered as content bearing it might be problem—because it is stop-word. The good thing about ***tf-idf*** is that it relies on both *term frequency (tf)* and *inverse* document *frequency (idf)*. This makes simple to reduce rank of common terms throughout the whole document corpus, but increase in rank of terms exist in fewer documents more frequently [22].

The ***term frequency*** (***tf***) is simply the number of times a given term appears in that document. This count is usually normalized to prevent a bias towards longer documents (which may have a higher term count regardless of the actual importance of that term in the document) to give a measure of importance of the term within particular document.

$$tf_{ij} = f_{ij} / max\{f_{ij}\} \quad \text{.....Equation 2. 1}$$

The ***inverse document*** frequency ***(idf)*** is measure whether the term is common or rare across all documents. It is obtained by dividing the total number of documents by the number of documents containing the term, then taking the logarithm and quotient.

Higher idf value is obtained for rare terms whereas lower value for common terms. It is mainly used to discriminate importance of term throughout the collection.

$$idf = \log_2 (N / df_i) \quad \text{.....Equation 2. 2}$$

Document frequency (*df*) - number of documents containing the given term. The more a term *t* occurs ***throughout*** all documents, the more poorly *t* discriminates between documents. The less frequently a term appears in the whole collection, the more discriminating it is.

Then the tf****idf*** is ***product*** of *tf* and *idf*.

$$tf*idf = tf_{ij} * \log_2 (N / df_i) \quad \text{.....Equation 2. 3}$$

2.5 Retrieval Models

A retrieval model specifies the details of the document representation, the query representation, and the matching function. There are two main reasons for having retrieval models. Primarily retrieval models guide research and provide the means for academic discussion. The second reason why we need retrieval model is because it serves as a blue print to implement an actual retrieval system[30]. An Information Retrieval models predicts and explains what a user will find relevant given the users query. Evaluation and experiments proves correctness of what it predicts by use of the model[7].

There are different categories of information retrieval models. The major categories are, but not limited to: the Boolean Model the Statistical Model (vector space model, probabilistic model), and the Linguistic and Knowledge-based Models. There is also Google's PageRank algorithm which uses web mining techniques [7].

2.5.1 Standard Boolean Model

Boolean model is the oldest model of information retrieval. In the Boolean there are three basic logical operators AND, OR and NOT. AND is logical product, OR is logical sum and NOT is logical difference. AND is used to group set of terms in to single query/statement. For example **'Information** AND **Technology'** is two term query combined by **'AND'.** In such case only document indexed with both terms will be retrieved. If terms in the user query are linked by operator OR, documented with either of terms or all terms will be retrieved. For example, if query is **Information** OR **Technology,** document containing Information, or Technology, or Information Technology will be retrieved[23][7].

What makes Boolean model good model is that it creates a sense of control to expert/user over the system. It is the user who is in charge for deciding what should or shouldn't be retrieved. Query reformulation is also simple because user is in charge of deciding what should be retrieved and should not. In contrast Boolean model may not retrieve anything if there is no matching document or, retrieves all documents if terms in query are matching with it. So there is no relevance judgment and ranking mechanism [6].

2.5.2 Statistical Model

The *vector space* and *probabilistic* models are the two major examples of the statistical retrieval approach. Both models use statistical information in the form of term frequencies to determine the relevance of documents with respect to a query. Although they differ in the way they use the term frequencies, both produce as their output a list of documents ranked by their estimated relevance. The statistical retrieval models address some of the problems of Boolean retrieval methods, but they have disadvantages of their own.

2.5.3 Vector Space Model

Vector space model is representation of index terms and query as vectors embedded in a high dimensional Euclidean space, where each terms is assigned as a separate dimension.

$$d_j = (w_{1j}, w_{2j}, \dots w_{tj})$$

$$q = (w_{1q}, w_{2q}, \dots w_{tq})$$

…… Equation 2. 4

The whole VSM involves three main procedures. The first is *indexing* of the document in the way that only content bearing terms represent the document. The second is *weighting* the indexed terms to enhance retrieval of relevant document. The final step is ranking the documents to show best matching with respect to the provided query by user [6].

i. Document Representation

Single document has many words, but it is only few words that describe content of the specific document. For instance prepositions and personal pronouns are non-content bearing terms. The main purpose of indexing is representing documents with only content bearing terms which have power to identify one document for the other [23]. Indexing can be based on term frequency where threshold is used to limit both high and low frequency terms. In order to make simpler, stop words are removed primarily from document and it will not be considered for representation.

ii. Term Weighting and similarity measurement

Term weighting is highly related to recall and precision. As in vector space model term weighting based on single term statistics throughout the entire document. In order to calculate term weighting we need term frequency, document frequency and length normalization. Product of these three main factors is result of term weighting.

Term frequency is occurrence of a term in a given document. It can be used as content descriptive for documents and generally used for basis of a weighted document vector [6]. In section 3.5 the detailed discussion is made on the VSM.

2.5.4 Probabilistic Model

The probabilistic retrieval model is based on the Probability Ranking Principle, which states that an information retrieval system is supposed to rank the documents based on their probability of relevance to the query, given all the evidence available. The principle takes into account that there is uncertainty in the representation of the information need and the documents. There can be a variety of sources of evidence that are used by the probabilistic retrieval methods, and the most common one is the statistical distribution of the terms in both the relevant and non-relevant documents [24][30].

The most familiar probabilistic model technique is Bayesian inference networks. Let me give highlight on how it rank documents by using multiple sources of evidence to compute the conditional probability P(Info need|document) that an information need is satisfied by a given document. An inference network consists of a directed acyclic dependency graph, where edges represent conditional dependency or causal relations between propositions represented by the nodes. The inference network consists of a document network, a concept representation network that represents indexing vocabulary, and a query network representing the information need. The concept representation network is the interface between documents and queries. To compute the rank of a document, the inference network is instantiated and the resulting probabilities are propagated through the network to derive a probability associated with the node representing the information need. These probabilities are used to rank documents [30].

The statistical approaches have the following strengths: Firstly they provide users with a relevance ranking of the retrieved documents. Hence, they enable users to control the output by setting a relevance threshold or by specifying a certain number of documents to display. Secondly queries can be easier to formulate because users do not have to learn a query language and can use natural language. Thirdly the uncertainty inherent in the choice of query concepts can be represented.

However, the statistical approaches have the following shortcomings: They have a limited expressive power. For example, the NOT operation cannot be represented because only positive weights are used. For example, the very common and important Boolean query ((A and B) or (C and D)) cannot be represented by a vector space query. Hence, the statistical approaches do not have the expressive power of the Boolean approach. The statistical approach also lacks the structure to express important linguistic features such as phrases. Proximity constraints are also difficult to express, a feature that is of great use for experienced searchers. The computation of the relevance scores can be computationally expensive. A ranked linear list provides users with a limited view of the information space and it does not directly suggest how to modify a query if the need arises. The queries have to contain a large number of words to improve the retrieval performance. As is the case for the Boolean approach, users are faced with the problem of having to choose the appropriate words that are also used in the relevant documents [7][31].

2.5.5 Linguistic and Knowledge-based Approaches

In the simplest form of automatic text retrieval, users enter a string of keywords that are used to search the inverted indexes of the document keywords. This approach retrieves documents based solely on the presence or absence of exact single word strings as specified by the logical representation of the query. Clearly this approach will miss many relevant documents because it does not capture the complete or deep meaning of the user's query.

Linguistic and knowledge-based approaches have been developed to address this problem by performing a morphological, syntactic and semantic analysis to retrieve documents more effectively [31]. In a morphological analysis, roots and affixes are analyzed to determine the part of speech (noun, verb, adjective etc.) of the words. Next complete phrases have to be parsed using some form of syntactic analysis. Finally, the linguistic methods have to resolve word ambiguities and/or generate relevant synonyms or quasi-synonyms based on the semantic relationships between words. The development of a sophisticated linguistic retrieval system is difficult and it requires complex knowledge bases of semantic information and retrieval heuristics. Hence these systems often require techniques that are commonly referred to as artificial intelligence or expert systems techniques [31].

DR-LINK Retrieval System: DR-LINK system represents an exemplary linguistic retrieval system. DR-LINK is based on the principle that retrieval should take place at the conceptual level and not at the word level. It attempts to retrieve documents on the basis of what people mean in their query and not just what they say in their query. DR-LINK system employs sophisticated, linguistic text processing techniques to capture the conceptual information in documents [23].

2.6 Related Works

From reviewed literatures the researches made sure that there is no work done on the specific area. No text retrieval system is developed for Afaan Oromo. But some scholars made attempt to enable cross-lingual IR system for Afaan Oromo rather than developing system that works for specifically Afaan Oromo. In the following we have reviewed some works has been done so far on the related area [32][33].

2.6.1 IR Systems for International Languages

These days information retrieval system is highly connected with human daily activities. There are many search engines for searching text documents, video, audio, software, and pictures. Additionally there are special purpose search engines like Amazon, which specifically works for online marketing of products and services. There are many works being done on the area of information retrieval in western world. But most of the work being done is for major technology languages like, English and French.

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

The writing system of an Arabic language uses Arabic alphabets [36]. 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. Additionally, available methodologies are difficult to implement for Arabic system. But it is agreeable on the importance of having IR system even when there are challenges.

2.6.2 IR Systems for Local Languages

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

2.6.3 Amharic Retrieval Systems

One of work done for Amharic retrieval is Amharic text retrieval system which is done by using latent semantic index(LSI) with regular value decomposition by Tewdros G, [14], at Addis Ababa University School of Information Science.

Tewodros [14], initiated to develop Amharic text retrieval using LSI technique. 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. In the paper stop-word list identified by Nega [37], was used. 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 is 0.6913 and by LSI 0.7157. LSI improves the VSM result by 2.4%. The future direction stated by the author includes: stemmed index terms might improve performance of the system and recommended to be used in the further work, if relevance feedback is supported performance of the system will be improved, and the algorithm LSI may record better performance in cross lingual(Amharic-English) if it is used. [14]

There are also work done on the specific area is the Application of hierarchical document cluster based information retrieval system for Amharic documents by Yalemsew [15]. Problem initiate includes: lack of system that enables access to documents based on their similarity, replacing existing manual system of retrieving documents which is very time consuming, and enabling browsing through document space. The main aim of the work is developing retrieval system that uses hierarchical document clustering for organizing, browsing, searching, and retrieval of Amharic web document. Document corpus used is collected from Walta Information Center (WIC), one of major news agency in Ethiopia. Files are stored on MS Access database and Visual Basic (VB) is used for programming. Techniques used in the paper are: indexing (normalization, stop word removal, stemming), term weighting (tf*idf), VSM is used for measuring similarity, and Frequent Item Based Hierarchical clustering (FIHC) is used for the clustering. Challenges that the researcher faced includes: absence of standard stop-word list, and ቃ and ሪ Amharic characters are not displayed correctly. In the paper there is no clearly expressed registered performance. The performance of the system is increased by 5% by use FIHC (still there is no clear statement about with what the author compared his work.). The future directions stated in the paper are: developing standard relevance judgment metrics, testing efficiency and effectiveness by comparing other algorithms, the work in the paper

supposed to be dynamic whereas it is not; so it needs further work to make it dynamic, and finally this work is done on desktop; if comparative study is done by using webserver it may give better result.

There are also works done in developing Amharic search engine. One of the study made on Amharic search engine is the attempt to design and implement Amharic Search engine by Tessema [13]. There was no search engine developed for Amharic language, as result most of commercial search engines are developed primarily in English. Search engine/IR system is language dependent (i.e. system that developed English doesn't necessarily work for Amharic). That is why the researcher focused on designing and implementation of the Amharic search engine. In the paper architecture developed for Amharic search engine. The architecture has three main components: Crawler, the Amharic Indexer, and Amharic Query Engine. It has been designed in the way that it can handle need of Amharic language users. The Crawler is responsible for collecting only Unicode encoded Amharic language documents from the Web. The Indexer component builds indexes from documents that it gets from the Crawler. It also tokenizes, removes stop words, apply stemming to the words before it indexes them. This index is used by the Query Engine to satisfy user needs. In the crawling technique n-gram based technique applied for identifying languages, because documents on the web are not only Amharic. In the indexing process tokenization, stop word removal, and stemming are included. Query engine has text box helps to enter user query in the Amharic character. Entered query passed under the same pre-processing that document did. Performance registered in this work is 99% precision and 52% recall. The average precision for top 10 ranked results is 75%. The future direction stated by the author is adding some more additional features as to increase effectiveness and efficiency of the system [13].

2.6.4 Oromo Cross Lingual information retrieval System

One of the studies done on bi-lingual retrieval of Oromo documents is the evaluation which is done in 2006 in CLEF (Conference and Labs of Evaluation Forum)[32]. In this study Oromo-English Cross lingual information retrieval was tested.

The study used dictionary based approach of CLIR in order to translate Oromo topics in to English query bag of words. The main purpose for the system is enabling Afaan Oromo users to access English documents, but by using Afaan Oromo query. The experimentation made in three phases.

In the experimentation query translated from Afaan Oromo in to its equivalent English first. The translation is done based on Machine Readable Dictionary (MRD). Additionally the English test date set used at CLEF was gained from two newspapers. The Afaan Oromo-English bilingual dictionary which was available in print format, and changed in to machine readable format by Optical Character Recognition (OCR) technology. That makes easier translation of Oromo queries in the English one. In order to remove stop words, list of stop word in Afaan Oromo was identified from Oromo text corpus. These identified stop words removed from query by the system before translation. Also stemming is done for as to remove conflate word variants in to the same stem root. Indexing and retrieval was done on Lucene an open source text search engine.

The other study made in 2008 [33], at CLEF as continuum part the previous one, to assess overall performance of the dictionary based approach. With this study significant result is obtained and it uncourageous.

The latest one is study done at Addis Ababa University on the master thesis 2011[38]. This study different from other research done on the specific area in the way its approach is corpus based approach, while other study used Dictionary Based Approach[32][33]. The holy Bible books and other sources gained from Regional Constitution of Oromiya Regional State are used as a parallel corpus. These documents are already translated and accessible in both English and Oromiffa. GIZA++ was used for word alignment and for text pre-processing. The experiment was done by using selected test documents and queries. Both English and Oromiffa queries are used in the experiment. The average precision of 0.468 for Afaan Oromo query and 0.316 for the English query was obtained at end of the experimentation.

Generally CLIR is used to enable retrieval of document in specific language with query in other language different from the language in which the document was written. But such systems may not consider the artefacts in the language, as a result of which they may not be as effective as possible to satisfy information need of users. So developing an IR system following modern IR principles that takes into account language related issues is necessary before designing a cross language retrieval system.

CHAPTER THREE

METHODS AND TECHNIQUE

Development of IR system involves various techniques and methods. Infromation retrieval sytem designed involves two main components that is : indexing and searching. The basic archituecture of information retrieval sytem is depicted in Figure 3.1.

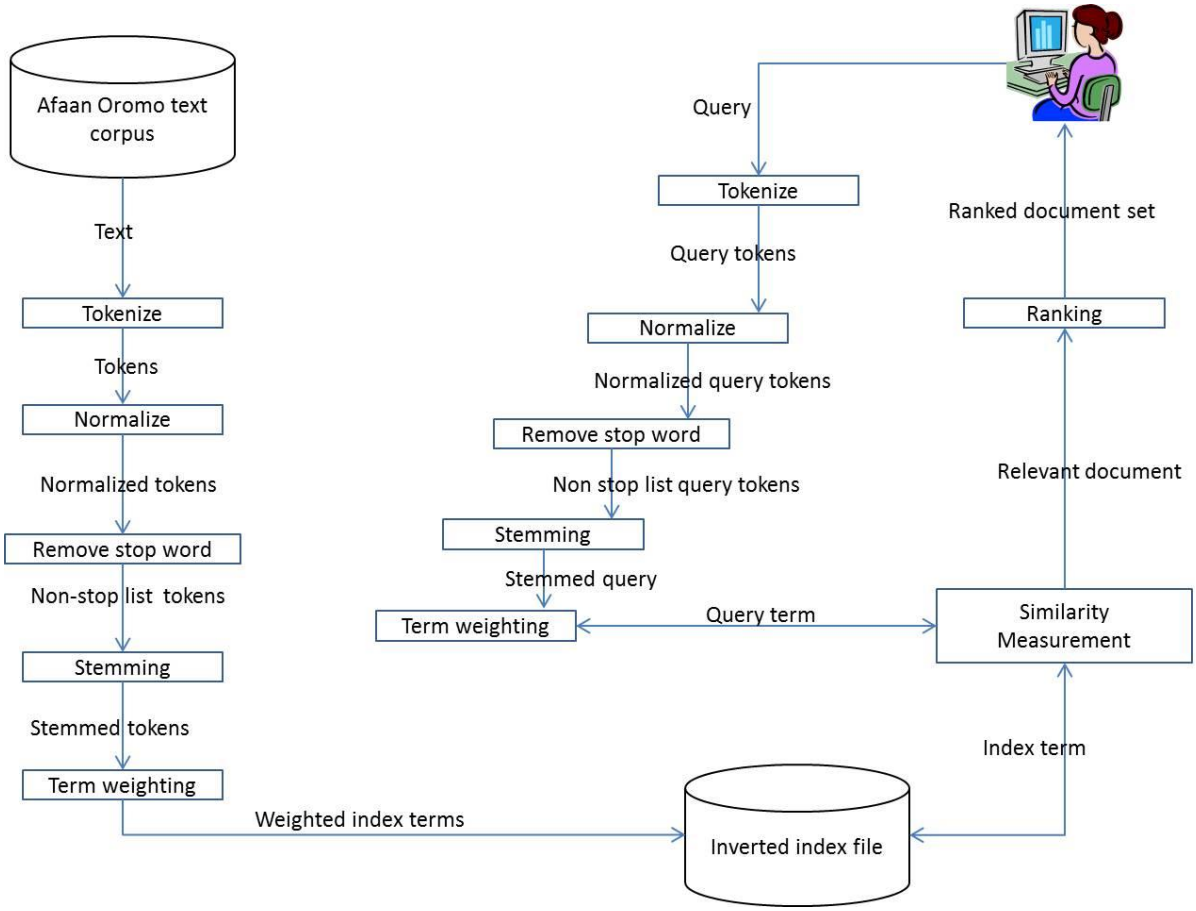


Figure 3. 1 Afaan Oromo Text Retrieval Sytem

Given Afaan Oromoo text corpus, the IR system orginize them using index file to enhance searching. The first step is tokenization of the text words to identify stream of tokens(or terms). This is followed by normilization in order to bring together similar word written with different punctuuation marks and variaiton cases(UPPER ,lower or mixed). The normilized token is checked as it is not stopword. Content bearing terms(non stop wors) are stemmed. For all stemmed tokens its respected weight calculted and then inverted index file is constructed.

On the searching similar text pre-processing(tokenization, norailization, stopword removal, and stemming) technique is followed as it was done in the indexing part. Then similarity is measurement techniques(cosine similarity) is used to retrieve and rank relevant documents.

3.1 Data Preprocessing and Corpus Preparation

In Information retrieval system corpus is needed for evaluation of the system. Document collected from different news articles and other online resources passes through different procedures in order to index and use the retrieval system.

Tokenization is the process of chopping character streams in to tokens, 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.

```
For file in corpus
     Define word delimiter to space
     Read files
          For file in read
          If there is  word delimiter
               Put each terms as separate token
```

Algorithm 3. 1 Tokenization

Normalization involves process of handling different writing system. Primarily every term in the document should be converted in to similar case format in this study lower case. For instance ‘IFORMATION’, ‘Information’, ‘information’ are all normalized to understandable as lower case ‘information’ the system. Secondly punctuation marks in all the documents will be omitted. Punctuations mark in Afaan Oromo are almost similar to that of English except apostrophe (“ ‘ ”) which is considered as part of word in Afaan Oromo. Some of punctuation mark are: ? : " ' ! | / ? @ # * ~ $ % ^ & ( ) { } < > [ ] _ + = - , ." ...\; - _ + £. Removal of punctuation marks helps to consider similar words with different punctuations mark, in similar way with no distinction of punctuation mark linked to it. For example ‘information?’, ‘information!’, ‘information.’, ‘information’, and etc. all are represented as ‘information’.

Stemmer algorithm going to be used for this work the one that is developed by Debela Tesfaye[39] which is developed, 2010. Debela's algorithm is rule based Afaan Oromo Stemmer. It is porter stemmer based algorithm for Afaan Oromo text document. Debela identified six stemmer rule clusters depending on the Afaan Oromo language grammatical rule [39].

Following is the algorithm developed to conflate word variants for Afaan Oromo text:

```
1. READ the next word to be stemmed
2. OPEN stop word file
     Read a word from the file until match occurs or End of File
     reached
     IF word exists in the stop word list
     Go to 5
     3. If word matches with one of the rules
          Remove the suffix and do the necessary adjustments
          Go  back to 3
     ELSE
          Go to 6
          4. Return the word and RECORD it in stem dictionary
     5. IF end of file not reached
          Go to 1
     ELSE
          Stop processing
     6. IF there is no applicable condition and action exist

          Remove vowel and return the result
```

Algorithm 3. 2 stemming algorithm

3.2 Inverted Index

An inverted file is a data structure for efficiently indexing texts by their words. One can view an inverted file as a list of words where each word is followed by the identifier of every text that contains the word. The number of occurrences of each word in a text is also stored in this structure. Even if users can choose data structure of his choice, what has to be included in the data structure is pre-determined by the technique. In this work inverted index is used for sake of indexing [40].

The major steps in creating an inverted index is first, collecting the document to be indexed, second, tokenize, normalize, remove stop list, stem and identifying list of tokens

to be indexed. At this stage language preprocessing should be done. Finally index file is created. Index file includes two files vocabulary file and post file.

The inverted file allows an IR system to quickly determine what documents contain a given set of words, and how often each word appears in the document. Other information can also be stored in the inverted file such as the location of each word in the text [40][41].

3.3 Vector Space Model(VSM)

The Vector Space Model (VSM) is a way of representing documents through the words that they contain as a vector. It is a standard technique in Information Retrieval system. The VSM allows decisions to be made about which documents are similar to each other and to keyword queries depending on similarity measurements. It is an algebraic model representing textual information as a vector. These vectors represents importance of a term in ***tf*idf*** weighting technique, and even absence or presence of the term in the document [23][42].

Each document is broken down into a word frequency table; that means in to inverted index data structure. When represented on vector space the tables are called vectors and can be stored as arrays of terms. A vocabulary is built from all the words in all documents in the system and it includes dictionary terms, which mean there is no repetition of terms. Each document is represented as a vector based against the vocabulary terms and query terms [7].

Vector space model procedure involves in to three stages. The first stage is the document indexing where content bearing terms are extracted from the document text. Following the indexing, weighting the indexed terms in order to enhance retrieval of document relevant to the user. The last stage is, ranking the document with respect to the query according to similarity measure.

Document indexing- automatic document indexing is used to create dictionary of terms simply selecting all terms from document and converting it in to a dimension in the vector space. The index file structure used in this study is 'inverted index' and it is discussed in section 3.3. Inverted file index has two files Vocabulary file and Post file. These files are used to build vectors of document versus terms.

Term weighting- the term weighting is discussed in section 3.4 of this paper. The main reason of computing term weight is to assigning weight depending on their level of importance. For every query terms in the query, term document vector is created. Then the weight will be calculated.

$$D_i = w_{d_{i1}}, w_{d_{i2}}, ..., w_{d_{it}}$$
$$Q = w_{q1}, w_{q2,}, ..., w_{qt} \quad \text{.....Equation 3. 1}$$
$$w = 0 \text{ if a term is absent}$$

The cosine similarity- There is many different ways to measure how similar two documents are to each other, or how similar a document is to a query. The cosine measure is a very common similarity measure. Using a similarity measure, a set of documents can be compared to a query and the most similar document returned. If the query term does not occur in the document, similarity measure score should be 0, else the more frequent the query term in the document, the higher the score (should be). Query and Document similarity is based on length and direction of their vectors. Here are basic facts in similarity [7][43].

- If d_1 is near d_2, then d_2 is near d_1.
- If d_1 near d_2, and d_2 near d_3, then d_1 is not far from d_3.
- No doc is closer to d than d itself.
- Distance between d_1 and d_2 is the length of the vector $|d_1 - d_2|$.

Let,

$$\begin{bmatrix} T_k = \text{term } k \text{ in document } D_i \\ tf_{ik} = \text{frequency of term } T_k \text{ in document } D_i \\ idf_k = \text{inverse document frequency of term } T_k \text{ in } C \\ N = \text{total number of documents in the collection } C \\ n_k = \text{the number of documents in } C \text{ that contain } T_k \\ idf_k = \log(n_k / N) \end{bmatrix}$$

First of all *tf*idf* weight is calculated for all query terms. In that process query is also considered as a document. That result gives weight of each document with respect to terms in the query. Finally the result obtained during term weighting will be used for further calculation of similarity measurement. .

Un-normalized similarity : $sim(Q, D_i) = \sum_{j=1}^{t} w_{qj} * w_{d_{ij}}$

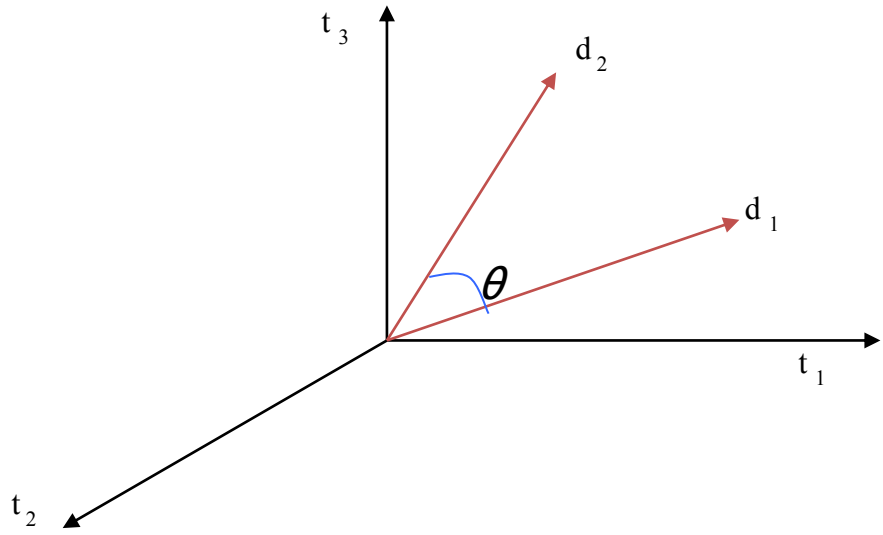
.....Equation 3. 2

Cosine similarity is normalized inner product.

cosine : $sim(Q, Di) = \frac{\sum_{j=1}^{t} w_{qj} * w_{d_{ij}}}{\sqrt{\sum_{j=1}^{t} (w_{qj})^2 * \sum_{j=1}^{t} (w_{d_{ij}})^2}}$

.....Equation 3. 3

To summarize distance between vectors d_1 and d_2 *captured* by the cosine of the angle *x* between them. For two vectors d and d' the cosine similarity between d and d' is given by:



Ranking the relevant document: the retrieved documents are ranked depending result from cosine similarity score. The document which is more similar with the query is ranked first other sorted depending on the score from very similar to least similar.

3.4 Evaluation of IR Performance

IR evaluation is highly related to the concept of "relevance". Relevance is the degree of correspondence between retrieved documents and information need of users. Since individuals need is varying relevance is subjective. Often peoples look relevance differently. What is relevant to somebody might be irrelevant to others. Nature of user query and document collection affects relevance. Additionally relevance depends on

individuals' personal needs, preference subject, level of knowledge, specialization, language, etc. [23].

Even if relevance is subjective concept there is no possibility of ignoring it. One of the approaches to deal with subjectivity is by generating "user profiles". User profile includes knowledge about user's needs, preferences, etc. This helps the IR system to give what 'meant' not what they asked for. To enhance this user profile should be generated automatically.

There are two way of measuring IR success, Precision and recall [6]. Precision is a ratio of relevant items retrieved to all items retrieved, or the probability of retrieved item is relevant. On the other way, Recall is ratio of relevant items retrieved to all relevant items in the corpus or the probability of relevant item retrieved. There is always trade-off between precision and recall. If every document in the collection is retrieved, it is obvious that all relevant documents will be retrieved, so that recall will be higher. In contrary when only little proportion of the retrieved document is relevant to given query, retrieving everything reduces precision (even to zero). The higher score in both recall and precision means the higher the performance of the system. [23].

Precision is the number of relevant documents a search retrieves divided by the total number of documents retrieved. In other word it is the fraction of the documents retrieved that are relevant to the user's information need.

$$\textbf{Precision} = \frac{|(\textbf{relevent document}) \cap (\textbf{retrieved document})|}{|(\textbf{retrieved documents})|} \quad \text{....Equation 3. 4}$$

Recall is the number of relevant documents retrieved divided by the total number of existing relevant documents that should have been retrieved. It is the fraction of the documents that are relevant to the query that are successfully retrieved.

$$\textbf{Recall} = \frac{|(\textbf{relevent document}) \cap (\textbf{retrieved document})|}{|(\textbf{relevant documents})|} \quad \text{.....Equation 3. 5}$$

Users' looks for their need satisfied; they don't need to get pile of documents in the corpus. The other problem is that relevant documents exist often redundantly. Beside to the fact that there is trade-off between precision and recall some people prefers higher recall, still others prefers higher precision. So, rather than considering precision and recall separately it is better to specify relative importance of both. Van Rijsbergen [6] designed

formula for Effectiveness, E-measure, where *P* is *precision*, *R* is *recall*, and α is non-zero parameter.

$$\mathbf{E} = \mathbf{1} - \frac{\mathbf{1}}{\propto\left(\frac{\mathbf{1}}{\mathbf{p}}\right)+(\mathbf{1}-\propto)\frac{\mathbf{1}}{\mathbf{R}}} \qquad \text{……..……Equation 3. 6}$$

Measuring precision is relatively easy; if documents are primarily defined as relevant and non-relevant. Of there should be settled threshold to reduce size, if the documents are ranked, e.g. only consider the top 100, or the top 20. Ranking often follows statistical or probabilistic in order to identify likelihood that documents are relevant.

Computing precision, recall and F-measure is done with unordered sets of documents. In order to evaluate ranked set of document the top ***n*** ranked documents are used. In this context Interpolated mean is used. The result obtained from the interpolated mean can be used for plotting *precision-recall curve.*

$$\mathbf{P\ interp(r)} \quad = \mathbf{max_{r1 \geq r}(r)} \qquad \text{……..Equation 3. 7}$$

Average Precision- precision and recall are single valued metrics based on the whole list of documents returned by the system. But for documents ranked by the system other mechanism needed.

While p(r) is precision at a rank of r a

$$\mathbf{AveP}= \int_0^1 \boldsymbol{p(r)dr} \qquad \text{……Equation 3. 8}$$

Where K is the rank in the sequence of retrieved documents, n is the number of retrieved documents, P(k) is the precision at cut-off K in the list, and Δr(k) is the change from items K-1 to K.

$$\text{AveP}=\sum_{k=1}^{n} \boldsymbol{P(k)\Delta r(k)} \qquad \text{…….……Equation 3. 9}$$

Generally

$$\mathbf{AveP}=\frac{\sum_{k=1}^{n}(\boldsymbol{P(k)\Delta rel(k)})}{\boldsymbol{number\ of\ relevent\ documents}} \qquad \text{……..Equation 3. 10}$$

Depending on the above listed evaluation techniques the experiment is conducted. In chapter 4 the experiment done is discussed.

CHAPTER FOUR

EXPERIMENTATION AND DISCUSSIONS

The problem behind this study is coming up with retrieval system that searches relevant documents from Afaan Oromo text that satisfies information need of users.

This study experimentally tested. The performance of IR system using corpus collected from VOA (Voice of America) Afaan Oromo service, www.Oromiya.com, online educational resources and other resources sources in Afaan Oromo available on the web. These articles involve different subjects like politics, education, culture, religion, history, social, health, economy and other events. For this study 100 different textual documents were collected from different news media

4.1 Index Construction

The prototype system developed is integrated from indexing and searching component. Both searching and indexing are built up of different sub components. Indexing involves tokenizing, normalization, stop word removal, stemming, and creating inverted file structure which in turn includes vocabulary file and posting file.

The code 4.1 is fragment code that tokenizes and normalizes the terms in the document.

```
characters = ".,!#$%^&*();:\n\t\\\"?!{}[]<>0123456789"
def tokenize(document):
    terms = document.lower().split()
    return [term.strip(characters) for term in terms]
```

Code 4. 5 Tokenization and normalization

Here this fragment code splits document based on space between each words, then convert every words 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 normalized and tokenized document will be returned for next process.

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.

Stop word removal

```
s=open('stopword.txt','r')
os.chdir('corpus')
stoplist=s.read()
s.close()
for i in token:
            if i  not in stoplist:# Stop list removal
                stemmed=thestemmer(i)
```

Code 4. 6: Stop word removal

Afaan Oromo stop word list is saved in text file `'stopword.txt'`. Some of Afan Oromo stop words are listed in Appendix III. 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.

```
if measure(token)>=1:
        if token.endswith('tii'):
            if token.endswith('ttii'):
                if token.endswith('ittii'):
                    stem=token.rstrip('ittii')
                    return stem
                stem=token.rstrip('ttii')
                return stem
            stem=token.rstrip('tii')
            return stem
        if token.endswith('f'):
            if token.endswith('if'):
                if token.endswith('iif'):
                    stem=token.rstrip('iif')
                    return stem
                stem=token.rstrip('if')
                return stem
            stem=token.rstrip('tii')
            return stem
     elif…
```

Code 4. 7 Stemming

The fragment code in Code 4.3 is creates stemmed words depending rule based stemming algorithm by Debela [39]. The algorithm checks first measure of the token, them implements rules of stripping inflections of words.

The next part is creating inverted file structure. The inverted file has two separates files vocabulary and posting file as it is discussed in section 3.2 and the Table 4.1 and 4.2 also depicts structure of inverted file (vocabulary file and posting file). The python code that generates these two files is available in Appendix IV.

Vocabulary files		
Terms	***Doc frequency***	***Collection frequency***
Aadaa	2	3
Misooma	1	2
Diinagdee	1	1
Fayyaa	3	4
Eegumsa	1	2

Table 4. 5 Vocabulry file

Posting file		
DocId	***Term frequency***	***Location***
Doc1	2	121,1212
Doc5	1	234
Doc2	2	123,434
Doc2	1	3897
Doc3	2	222,232
Doc4	1	545
Doc5	1	765
Doc1	2	3433,3443

Table 4. 6 Posting file

The searching component is also written with a python code which helps to implement vector space model. Query text pre-processing is done in similar way to indexing part. Pre-processing part involves tokenization, normalization, stop word removal and stemming. Each query term should pass through each of these processes as it was elaborated in Figure 3.1 on the architecture of Afaan Oromo text retrieval system.

4.2 Searching and VSM Construction

The searching component of the prototype uses VSM model. Users' information need is represented as query. The detail of query term selection is discussed in section 4.3. Here is in this component similarity measurement and ranking is done. From the actual code used in the prototype some of the code is posted for explanation of the system.

The vocabulary file holds each unique non stop-word tokens in the documents with its document frequency and collection frequency. The document frequency and collection frequency is cross referenced to posting file. Posting file holds term terms with their term frequency, positions in the documents and name of each document holding the

term. So the search algorithm brings together these things in order to calculate similarity and rank documents. The inverted file index has computed weight terms in the document and weight for the query is also need to be computed.

```
    f=open('vocabulary.txt','r') #open vocabulary file for reading
    ff=open('post.txt','r') #open posting file for reading
          ...
     dic={}
         for x in y:
             z=x.split()
             df=z[1]
             term1=z[0]
             st=''
             for xx in yy:
                 zz=xx.split()
                 term2=zz[0]
                 doc=zz[1]
                 tf=zz[2]
                 if term1==term2:
                     doc=zz[1]
                     wf=wordfreq(doc)
                     tf1=float(tf)
                     tf1=tf1/wf
                     dfi=float(df)
                     df1=N/(dfi+0.000000001)
                     p=math.log(df1,2)
                     idf=tf1*p
                     st=doc+' '+term2
                     dic[st]=idf
         return dic
```

Code 4. 8 Connecting vocabulary and posting files in order to calculate term frequency

The inverted index file (vocabulary file and posting file) which is demonstrated with example on Table 4.1 and Table 4.2 is two separate file, but used together for term weighting and similarity measurement. So in order to implement VSM model the vector of query versus document is created by use of these files. After all it is easier to retrieve relevant documents by depending on level of their relevance by using cosine similarity. Cosine similarity is discussed section 3.3 under vector space model. The performance of the system is tested by using user information need queries.

4.3 Query selection

In order to make the experiment nine (9) queries are identified. These queries are marked across each document as either relevant or irrelevant to make relevance evaluation. The main importance of having identified queries is to evaluate the

performance of the system. These queries are selected subjectively by the research after reviewing content each article manually.

After all queries and documents should be stored in the way it can be used for term weighting and application of VSM. So document and queries are arranged as a document and query as term-document matrix. It includes term frequency (*tf*), collection frequency (*cf*), document frequency (*df*). In Table 4.3, summary of relevance judgement for selected 9 queries is listed and the detailed list available in Appendix II. According to that the relevance judgement each document is identified to be as a relevant or non-relevant toward each query terms.

Queries		**Relevant**	**Non relevant**
Q1	*Beela gaafa afrikaa*	**7**	**93**
Q2	*Eebba Waaqaa*	**3**	**97**
Q3	*Mirga dhala namaa*	**21**	**79**
Q4	*Dhuma biyya lafaa*	**14**	**86**
Q5	*Sirna Gadaa*	**12**	**88**
Q6	*Ayyaana Irressaa*	**7**	**93**
Q7	*Eegumsa Fayyaa*	**8**	**92**
Q8	*Aayyaanaa fi Ayyaantummaa*	**5**	**95**
Q9	*Shaampiyoonaa tapha guutuu Itiyoophiyaa*	**6**	**94**

Table 4. 7 Relevance judgment summary

4.4 System evaluation method

Once after IR system is designed its performance and accuracy should be evaluated. Evaluation of IR system involves two things effectiveness and efficiency [24]. Even if it is important to evaluate both effectives and efficiency, as the objective of this study the researcher considered as a vital part of the documentation only effectiveness. The IR system effectiveness evaluated in various ways. The main one are precision and recall[23]. In this study F-measure and Interpolated average precision is also used.

Precision is the number of relevant documents a search retrieves divided by the total number of documents retrieved. In other word it is the fraction of the documents retrieved that are relevant to the user's information need. Recall is the number of relevant documents retrieved divided by the total number of existing relevant

documents that should have been retrieved. It is the fraction of the documents that are relevant to the query that are successfully retrieved.

4.5 Experimentation

In this experiment 10 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 4.4.

	Query terms	***Precision***	***Recall***
Q1	*Beela gaafa afrikaa*	1	0.167
Q2	*Eebba Waaqaa*	0.667	0.667
Q3	*Mirga dhala namaa*	0.323	0.733
Q4	*Dhuma biyya lafaa*	0.143	1
Q5	*Sirna Gadaa*	0.375	0.818
Q6	*Ayyaana Irressaa*	0.5	0.333
Q7	*Eegumsa Fayyaa*	0.667	0.92
Q8	*Aayyaanaa fi Ayyaantummaa*	0.5	0.83
Q9	*Shaampiyoonaa tapha guutuu Itiyoophiyaa*	1	0.167
Average		***0.575***	***0.6264***

Table 4.8 Detailed result of performance

As it is shown in Table 4.4 the obtained result is 0.575(57.5%) precision and 0.6264(62.64%) 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 57.5% relevant documents for these given nine (9) queries. This is because of morphological inflection nature of Afaan Oromo writing system. Afaan Oromo terms are highly inflected for number, genders, possession, plural, postpositions, and conjunctions. The stemming algorithm should handle that. But from the experimentations we could understand that the stemming algorithm is not working well to handle this problem. Here are common examples of endings of root words

which create inflection. plural makers *(-oota, -oolii, -wwam, -lee, -an, -een, -eetii, eeyyi, -ii,* etc) , gender makers(*-ttii, -tuu, -eessa, -a, itti*, etc.) verbs(*–uu, -a, -na, -ti, -an* and etc.), possession (*–ke, -ko,-sa,*) postpositions (*-bira, -dura,-irra,-jala, akka, gara, gad, -n, -dha, -jira*) conjunctions,*(-fi, -lee,)* article (*-moo, -icha, -itti,)* . There are many terms which have the same meaning and used interchangeably in writing systems. But there is no mechanism is used to manage polysemy and synonymy in this study. The other thing is the case of synonymy words.

The Quer1 (*Beela gaafa afrikaa*) registered better precision but lower recall. In this case all relevant documents are retrieved but there irrelevant documents which are retrieved too. This is because the term '*gaafa'* has many meaning and different documents.

There are many different ways of writing in Afaan Oromo. For Query6 (*Ayyaana Irressaa*.) the registered performance is 33.3% recall and 50% precision. The reason is there are different ways of writing for the term '*Irressaa', '*which is Oromo thanks giving celebration'in English. It is written as '*Irreecha'*, '*Irreechaa'*, '*Irreensaa'*, '*Irreessa'* and '*Irreessaa*'. So this creates ambiguity and only few documents are retrieved. It should be handled by either stemming algorism or other mechanism.

Recall measures how many of retrieved documents are relevant. In this is study better recall is registered when compared to precision. 62.64% of the retrieved document those are relevant. This is because of synonymy and polysemy. For instance Query9 (*Shaampiyoonaa tapha guutuu Itiyoophiyaa)* is one of query which registered lower recall. In this case most of retrieved documents are non-relevant documents. This happens because '*Ithiyoophiyaa'* which is *Ehtiopia* in English *and 'guutuu' w*hich is *'full'* in English are available in many documents. That means many documents which are not relevent are retrieved. Query reformulation can handle this problem.

General the work done is registered promising performance. Better performance will be achieved if stemmer algorithm is improved, and there is mechanism to handle polysemy and synonymy.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

Text retrieval system is very important for retrieval of textual document. The study attempts to develop Afaan Oromo IR system. The developed prototype has two modules: indexing and searching. The indexing part of the work involves tokenizing, normalization, stop word removal and stemming. Indexing Afaan Oromo text document has common ground with that of the English langauge because both the language uses Latin alphabets. But Afaan Oromo indexing varies in many different ways. As it has been identified by the study Afaan Oromo has its own grammar (which is called 'Seerluga') from English. Tokenization of Afaan Oromo is almost similar to the English one except apostrophe (') is not punctuation mark in Afaan Oromo, rather it is part of words. Normalization of Afaan Oromo documents is also very important. It has both language dependent and language independent features. The Afaan Oromo stemmer was the only component adopted from previously conducted research. Stemming Oromiffa documents has its own unique procedures, totally different from the English one. Generally speaking, Afaan Oromo document indexing needs its own algorithm and procedure as it is different from other language.

The Afaan Oromo text retrieval system developed has also searching components. The main parts are query pre-processing, similarity measurement and ranking. Pre-processing is language dependent process that involves tokenizing, normalization, stop word removal and stemming. Whereas similarity measurement and ranking follows vector space model which is applicable for every language.

The retrieval model used for the study is VSM. According to the experimentation made the system registered 0.575 *precision* and 0.6264 recall which is promising to design Afaan Oromo information retrieval system that searches within large corpus. The main challenge in the study is the inability of the stemmer to efficiently handle word variants. Additionally synonymy and polysemy nature of Afaan Oromo language greatly affect the retrieval performance which initiate the integration of query operations.

5.2 Recommendations and future directions

Afaan Oromo retrieval system is has wide open place for future study. The area is on beginning level. It needs collaboration of researcher and funding organization. In fact we would like to recommend the followings to concerning body.

- 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.
- Stemming algorithm used in this study doesn't work well for every inflated words in the language. Additionally there are conditions which never addressed by the stemming algorithm. So there should be further study to come up with better stemming algorithm that works for Afaan Oromo documents.
- Query reformulation (query expansion) algorithm improves system performance by managing polysemy and synonymy words. So immediate research work should be conducted.
- The study is conducted for the first time by using VSM. It wasn't compared to any other information retrieval models. Further study is needed to figure out best model that works for Afaan Oromo retrieval system
- To handle polysemy and synonymy terms it is important to have thesaurus system. This is also very important area of the study.
- Nowadays ontology is the main area of study in information retrieval. It is believed that the system will have better performance if ontology based IR system is implemented for Afaan Oromo text retrieval system.
- This work is "text retrieval system" for Afaan Oromo 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.

Bibliography

[1] P. Ingwersen, *Information Retrieval Interaction*, 1st ed. London: Taylor Graham Publishing, 2002.

[2] R. Baeza-Yates, *Information Retrieval: Data Structure & Algorithms*, 1st ed. Waterloo: University of Waterloo, 2004, pp. 1-630.

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

[4] V. Bush, "As We May Think," *The Atlantic Monthly*, vol. 176, no. 1, pp. 101-108, 1945.

[5] Bruce R. Schatz, "Information Retrieval in Digital Libraries: Bringing Search to the Net," Journal of Science, pp. 327-334, vol. 275, 1997

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

[7] D. Hiemstra, Information Retrieval Models, Wiley Online. New york: John Wiley & Sons, Ltd, 2009.

[8] H. E. Wolff, "Cushitic Languages," *Encyclopedia Britannica*. Encyclopedia Britannica, Inc., 2012.

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

[10] Simon Ager, "Oromo Language," *Online edition*, 2012. [Online]. Available: www.sas.upenn.edu/African_Studies/Hornet/Afaan_Oromo_19777.html . [Accessed: 25-Feb-2012].

[11] Etnologue, "Show Language," *Online edition*, 2009. [Online]. Available: http://www.ethnologue.com/web.asp. [Accessed: 01-Mar-2012].

[12] A. Toffler, *The third wave*, 1st ed. Washington Dc: Progress & Freedom Foundation, 1980.

[13] Tessema Mindaye and Solomon Atnafu, "Design and Implementation of Amharic Search Engine," *5th international conference on signal image technology and internet based system*, 2009.

[14] Tewodros Hailemeskel Gebermariam, "Amharic Text Retrieval : An Experiment Using Latent Semantic Indexing (LSI) With Singular Value Decomposition (SVD)," *MSc Thesis*, School of Information Science, Addis Ababa University, 2003.

[15] Bizuneh Mamuye Birhan, "The Application of Websom for Amharic Text," *MSc Thesis*, School of Information Science, Addis Ababa University, 2003.

[16] Hassen Redwan, Tessema Mindaye, and Solomon Atnafu, "Search Engine for Amharic Web Content," *IEEE Africon*, vol. 978–1-4244, no. 09, pp. 1-6, 2009.

[17] C. R. Kothari, Research Methodology: Methods and Techniques, 2nd ed. New Delhi: New Age International Ltd., 2004.

[18] Python Software Foundation, "About Python," 2012. [Online]. Available: http://www.python.org/about/. [Accessed: 12-Jan-2012].

[19] Language Materials Project, "Afaan Oromo," 2010. [Online]. Available: http://www.lmp.ucla.edu/default.aspx?menu=001. [Accessed: 04-Mar-2012].

[20] 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.

[21] Mylanguage.org, "Oromo Numbers," 2011. [Online]. Available: http://www.mylanguages.org/learn_oromo.php. [Accessed: 12-Jan-2012].

[22] "Information Retrieval and Extraction 2008," *University of Washington*, 2008. [Online]. Available: http://nlg3.csie.ntu.edu.tw/courses/IR/2008midtermSolutions.htm. [Accessed: 28-Mar-2012].

[23] E. Greengrass, "Information Retrieval: A survey," *Information Retrieval*, vol. 163, no. November, pp. 141-163, 2000.

[24] A. Singhal, "Modern information retrieval: A brief overview," *IEEE Data Engineering Bulletin*, vol. 24, no. 3, pp. 35-43, 2001.

[25] A. Das and A. Jain, "Indexing the World Wide Web: The Journey So Far," *Google Inc, USA*, vol. 1, no. 1, pp. 1-24, 2008.

[26] J. Etzold, A. Brousseau, P. Grimm, and T. Steiner, "Context-aware Querying for Multimodal Search Engines," *Springer-Verlag Berlin Heidelberg*, vol. 6, no. 2012, pp. 728-729, 2011.

[27] Y. Fang, N. Somasundaram, L. Si, J. Ko, and A. P. Mathur, "Analysis of An Expert Search Query Log Categories and Subject Descriptors," *Symposium A Quarterly Journal In Modern Foreign Literatures*, vol. 64, no. 18, pp. 1189-1190, 2011.

[28] Y. Y. Yao, Information Retrieval Support Systems. Boca Raton: Taylor & Francis Group, 2012, pp. 1-778.

[29] P. Soucy, "Beyond TFIDF Weighting for Text Categorization in the Vector Space Model," *International Joint Conference on Artificial Intellegence*, vol. 1, no. 1, pp. 1-6, 2005.

[30] N. Fuhr, "Probabilistic Models in Information Retrieval," *SpringerLink*, vol. 11, no. 3(2008), pp. 251-265, 2008.

[31] H. R. Turtle and W. B. Croft, “A Comparison of Text Retrieval Models,” *Computer*, vol. 35, no. 3, pp. 279-290, 1992.

[32] Kula Kekeba, P. Pingali, and V. Varma, “Hindi , Telugu , Oromo , English CLIR Evaluation,” *Springer-Verlag Berlin Heidelberg*, vol. CLEF 2006, LNCS 4730, pp. 35-42, 2007.

[33] Kula Kekeba, V. Varma, and P. Pingali, “Evaluation of Oromo-English Cross-Language Information Retrieval,” *International Joint Conference on Artificial Intelligence (IJCAI)-2007*, vol. IIIT/TR/20, June, 2008.

[34] M. P. Lewis, “Statistical Summaries,” *16th edition*, 2009. [Online]. Available: http://www.ethnologue.com. [Accessed: 05-Jun-2012].

[35] B. H. Juang and L. R. Rabiner, “Modern Information Retrieval – A Brief History of the Technology Development,” pp. 1-24.

[36] K. Aloufi, “Diacritic Oriented Arabic Information Retrieval System,” *International Journal of Computer Science and Security*, vol. 5, no. 1, pp. 1-67, 2011.

[37] Nega Alemayehu and P. Willett, “Stemming of Amharic Words for Information Retrieval,” *American Society for Information Science*, vol. 50, no. 6, pp. 524-529, 2002.

[38] 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.

[39] Tesfaye Debela and Ermias Abebe, “Designing a Rule Based Stemmer for Afaan Oromo Text,” *International journal of Computational Linguistics*, Addis Ababa,vol. 1, no. 2, pp. 1-11, 2010.

[40] C. Monz and M. de Rijke, Inverted Index Construction. Christof Monz & Maarten de Rijke, 2002, pp. 1-39.

[41] S. Heinz, “Efficient single-pass index construction for text databases,” *Journal of the American Society for*, vol. 54, no. 8, pp. 713-729, 2003.

[42] J. Becker, “Topic-Based Vector Space Model,” *Bussiness Information Systems*, vol. BIS2003, pp. 1-6, 2003.

[43] R. Wilkinson and J. Zobel, “Similarity Measures for Short Queries,” *NIST Special Publication*, vol. 3, no. 5, pp. 1-26, 1996.

Appendix I- Qubee fi Dubbiftuu (Qubees and their phone)

Sagalee gaggabaaboo (Short sounds)				
`a	`e	`i	`o	`u
ba	be	bi	bo	bu
ca	ce	ci	co	cu
cha	che	chi	co	cu
da	de	di	do	du
dha	dhe	dhi	dho	dhu
fa	fe	fi	fo	fu
ga	ge	gi	go	gu
ha	he	hi	ho	hu
ja	je	je	jo	ju
ka	ke	ki	ko	ku
la	le	li	lo	lu
ma	me	mi	mo	mu
na	ne	ni	no	nu
nya	nye	nyi	nyo	nyu
pa	pe	pi	po	pu
pha	phe	phi	pho	phu
qa	qe	qi	qo	qu
ra	re	ri	ro	Ru
sa	se	si	so	su
sha	she	shi	sho	shu
ta	te	ti	to	tu
va	ve	vi	vo	vu
wa	we	wi	wo	wu
xa	xe	xi	xo	xu
ya	ye	yi	yo	yu
za	ze	zi	zo	zu

Sagalee dhedheeroo (Long sounds)				
`aa	`ee	`ii	`oo	`uu
baa	bee	bii	boo	buu
caa	cee	cii	coo	cuu
chaa	chee	chii	coo	cuu
daa	dee	dii	doo	duu
dha	dhe	dhi	dho	dhu
faa	fee	fii	foo	fuu
gaa	gee	gii	goo	guu
haa	hee	hii	hoo	huu
jaa	jee	jee	joo	juu
kaa	kee	kii	koo	kuu
laa	lee	lii	loo	luu
maa	mee	mii	moo	muu
naa	nee	nii	noo	nuu
nyaa	Nyee	nyii	nyoo	nyuu
Paa	pee	pii	poo	puu
Phaa	phee	phii	phoo	phuu
qaa	qee	qii	qoo	quu
raa	ree	rii	roo	ruu
saa	see	sii	soo	suu
shaa	shee	shii	shoo	shuu
taa	tee	tii	too	tuu
vaa	vee	vii	voo	vuu
waa	wee	wii	woo	wuu
xaa	xee	xii	xoo	xuu
yaa	yee	yii	yoo	yuu
zaa	zee	zii	zoo	zuu

Appendix II-Relevance Judgment Table

Text files	***Q1***	***Q2***	***Q3***	***Q4***	***Q5***	**Q6**	***Q7***	***Q8***	***Q9***	***Q10***
001.txt	NR	R	NR	NR	NR	NR	NR	NR	R	NR
002_Aadaa.txt	NR	R	NR	NR	NR	NR	NR	NR	NR	NR
003_adaaamantii.txt	NR	R	NR	NR	NR	NR	NR	NR	NR	NR
004_AfaanOromoo.txt	NR	NR	NR	R	NR	NR	NR	NR	NR	NR
005_Ayaantota.txt	NR	R	NR	R	R	NR	NR	NR	NR	NR
006_AyyaanaafiUumama.txt	NR	R	NR	R	NR	NR	NR	NR	NR	NR
007_Ayyaanotawaggaa.txt	NR	R	NR	NR	NR	NR	NR	NR	NR	NR
008_BiyyaOromoo.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
009_QubeeAfaanOromoobabal.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
010_Bulchiinsa.txt	NR	NR	R	NR	R	NR	NR	NR	NR	NR
011_Dhahaa.txt	NR	R	NR	R	NR	NR	NR	NR	NR	NR
012_EEBBAOROMO.txt	NR	R	NR	R	NR	NR	NR	NR	NR	NR
013_EebbaWaaqaa.txt	NR	R	NR	NR	NR	NR	NR	NR	NR	NR
014_Gadaa.txt	NR	NR	R	NR	R	NR	NR	NR	NR	NR
015_Jireenya.txt	NR	R	NR	NR	NR	NR	NR	NR	NR	NR
016_LammiiwwanAmeerikaaHoogganaaQabsooMirgaNamaaKabajaanYaadatan.txt	NR	NR	R	NR	NR	NR	NR	NR	NR	NR
017_nyaata.txt	NR	R	NR	NR	NR	NR	NR	NR	NR	NR
018_dhaqnaqabaa.txt	NR	NR	R	NR	NR	NR	NR	R	NR	NR
019_sirba.txt	NR	R	NR	NR	NR	NR	NR	NR	NR	NR
020_Taphoota.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
021_UmmataOromoo.txt	NR	R	NR	NR	NR	NR	NR	NR	NR	NR
022_Waaeeoromoo.txt	NR	R	NR	NR	NR	NR	NR	NR	NR	NR
023_Waaqeffachuu.txt	NR	NR	NR	R	NR	NR	NR	NR	NR	NR
024_Waaqeffannaa.txt	NR	NR	NR	R	NR	NR	NR	NR	NR	NR
025_Waaqeffannaa_Madda_Qulqulloominaa.txt	NR	NR	NR	R	NR	NR	NR	NR	NR	NR
027_YayyabaGWA.txt	NR	NR	NR	R	NR	NR	NR	NR	NR	NR
037_MAKMMAAKSSAOROMOO.txt	NR	R	NR	NR	R	NR	NR	NR	NR	NR
042_MAKMMAAKSSAOROMOO.txt	NR	R	NR	NR	NR	NR	NR	NR	NR	NR
045_namniyeroosaabeekuhinjiru.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
046_sifgaruujireenyijira.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
047_fayyaahaadholeefihijoollee.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
048_Tokenmaalii.txt	NR	R	NR	NR	R	NR	NR	R	NR	NR
110_Maalimaa.txt	NR	R	NR	NR	NR	NR	NR	NR	NR	NR
111_Ekerdubbiftuu.txt	NR	R	NR	R	NR	NR	NR	NR	NR	NR
112_MooraLaaltuu.txt	NR	R	NR	R	NR	NR	NR	NR	NR	NR
113_Sorsa.txt	NR	R	NR	R	NR	NR	NR	NR	NR	NR
114_Arba.txt	NR	R	NR	R	NR	NR	NR	NR	NR	NR
115_Gardaaduma.txt	NR	R	R	NR	NR	NR	NR	NR	NR	NR
116_Sonsa.txt	NR	R	R	NR	NR	NR	NR	NR	NR	NR

117_Basaaduraa.txt	NR	R	R	NR	NR	NR	NR	NR	NR	NR
118_Bitaaballoo.txt	NR	R	R	NR	NR	NR	NR	NR	NR	NR
119_Rubruma.txt	NR	R	R	NR	R	NR	NR	NR	NR	NR
120_balloo.txt	NR	R	R	NR	NR	NR	NR	NR	NR	NR
121_Ruuda.txt	NR	R	R	NR	NR	NR	NR	NR	NR	NR
122_Ayyaanacimdiiyknballoo.txt	NR	R	R	NR	NR	NR	NR	NR	NR	NR
123_Ayyaanota.txt	NR	R	R	NR	NR	NR	NR	NR	NR	NR
124_HaadholiifBiyyiJireenyaafGaariifiHamtuuTa.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
125_Keniyaaethiopiadjibuthidhannaa.txt	R	NR	NR	NR	NR	NR	NR	NR	NR	NR
126_KeenyaanJaraBuufataOtobusiiNaayroobiittiNamaFixeJetteeShakkiteAfurQabatte.txt	R	NR	NR	NR	NR	NR	NR	NR	NR	NR
127_DhukkubaGaraachaaFayyuunNiDanda.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
128_Gargaarsa.txt	R	NR	NR	NR	NR	NR	NR	NR	NR	NR
129_Beelasomaliyaa.txt	R	NR	NR	NR	NR	NR	NR	NR	NR	NR
130_beela_afrikaa.txt	R	NR	NR	NR	NR	NR	NR	NR	NR	NR
131_PaartiileeOromoo.txt	NR	NR	R	NR	NR	NR	NR	NR	NR	NR
132_Dubbisan.txt	NR	NR	R	NR	R	NR	NR	NR	NR	NR
133_HiyyummaaFiEtiyoophiyaa.txt	R	NR	NR	NR	NR	NR	NR	NR	NR	NR
134_SudaaniifiSudaanKibbaaWal.txt	NR	NR	NR	NR	NR	R	NR	NR	NR	NR
135_SudaanKibbaaBilisummaadhaaBoodaMaaltuEeggata.txt	NR	NR	NR	NR	NR	R	NR	NR	NR	NR
136_JECHOotaAFAANOROMOO.txt	NR	NR	R	NR	R	NR	NR	NR	NR	NR
136_WaaltaanMirgaDhala.txt	NR	R	NR	NR	NR	NR	NR	NR	NR	NR
137_IrreessiIrreefiMallattooTokkummaaKeenyaati.txt	NR	NR	R	NR	R	NR	R	NR	NR	NR
138_Irreecha.txt	NR	NR	NR	NR	NR	NR	R	NR	NR	NR
139_AYYAANAIRREECHAA.txt	NR	NR	R	NR	NR	NR	R	NR	NR	NR
140_GodaansaOromooJaarraa16ffaanduratti.txt	NR	NR	NR	NR	R	NR	NR	NR	NR	NR
141_BeeksisaGumiiWaaqeffannaaNorway.txt	NR	NR	NR	NR	R	NR	NR	NR	NR	NR
142_QayyabatatokkummaaOromoo.txt	NR	NR	R	NR	R	NR	NR	NR	NR	NR
143_AfrikaagamaGaafaaBaranalleeBokkaan.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
144_Oromoofimanneen.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
145_beelaagargaaruuf.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
146_vaayreesiiHIV.txt	NR	NR	NR	NR	NR	NR	NR	R	NR	NR
147_CarraanBiyyootaHiyyeeyyiiMaalTa.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
148_DhukkubaBusaafiRaammoo.txt	NR	NR	NR	NR	NR	NR	NR	R	NR	NR
149_Pooliyoo.txt	NR	NR	NR	NR	NR	NR	NR	R	NR	NR
150_Dhukkubiisaaynuusaayti	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR

s.txt										
151_GumiinWaa.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
152_GuyyaaDhukkubaBusaa SadarkaaAddunyaa.txt	NR	NR	NR	NR	NR	NR	NR	R	NR	NR
153_GuyyaanNarsoota.txt	NR	NR	NR	NR	NR	NR	NR	R	NR	NR
154_UmriinLakkobsa.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
155_BaraQaroomii.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
156_ijoolleeHorsiiseeBulaatiinbarsiisuu.txt	NR	NR	R	NR	NR	NR	NR	NR	NR	NR
157_Oromiyaatti.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
158_WaldaanMisoomaOromiyaa.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
159_Dua.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
160_FilannooPrezidaantummaaAmerikaaf.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
161_martinLuther.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
162_MagaalaaAmbootti.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
163_MiidhaanDubartii.txt	NR	NR	NR	NR	NR	NR	NR	R	NR	NR
164_Seenaan.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
165_Atileeti.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
166_Dhaamsa.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	R
167_BishaanDaakichaa.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	R
168_DorgommiinAtileetiksii.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	R
169_GareenAtileetiksii.txt	NR	NR	NR	NR	NR	NR	NR	NR	R	R
170_HanqinniBajataaAmeriikaaRaasaaJira.txt	NR	NR	NR	NR	NR	NR	NR	NR	R	NR
171_KomishiniinIspoortiiFederaalaa.txt	NR	NR	NR	NR	NR	NR	NR	NR	R	R
172_LigiiOromiyaa.txt	NR	NR	NR	NR	NR	NR	NR	NR	R	NR
173_Shaamiyonaan.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	R
174_WikiLeaks.txt	NR	NR	R	NR	NR	NR	NR	NR	NR	NR
175_OomishaOromiya.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
175_WaajjraaleenMootummaaodeeffannoo.txt	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR

Keys:

R- Relevant

NR- non-relevant

Appendix III- Afaan Oromo Stop Words

aanee
agarsiisoo
akka
akkam
akkasumas
akkum
akkuma
ala
alatti
alla
amma
ammo
ammoo
an
ana
ani
ati
bira
booda
booddee
dabalatees
dhaan
dudduuba
dugda
dura
duuba
eega
eegana
eegasii
ennaa
erga
ergii
f
faallaa
fagaatee
fi
fullee
fuullee
gajjallaa
gama
gararraa
garas
garuu
giddu
gidduu
gubbaa
ha
hamma
hanga
henna
hoggaa
hogguu
hoo
hoo
illee
immoo
ini
innaa
inni
irra
irraa
irraan
isa
isaa
isaaf
isaan
isaani
isaanii
isaaniitiin
isaanirraa
isaanitti
isaatiin
isarraa
isatti
isee
iseen
ishee
ishii
ishiif
ishiin
ishiirraa
ishiitti
ishiitti
isii
isiin
isin
isini
isinii
isiniif
isiniin
isinirraa
isinitti
ittaanee
itti
itumallee
ituu
ituullee
jala
jara
jechaan
jechoota
jechuu
jechuun
kan
kana
kanaa
kanaaf
kanaafi
kanaafi
kanaafuu
kanaan
kanaatti
karaa
kee
keenna
keenya
keessa
keessan
keessatti
kiyya
koo
kun
lafa
lama
malee
manna
maqaa
moo
na
naa
naaf
naan
naannoo
narraa
natti
nu
nu'i
nurraa
nuti
nutti
nuu
nuuf
nuun
nuy
odoo
ofii
oggaa
oo
osoo
otoo
otumallee
otuu
otuullee
saaniif
sadii
sana
saniif
si
sii
siif
siin
silaa
silaa
simmoo
sinitti
siqee
sirraa
sitti
sun
tahullee
tana
tanaaf
tanaafi
tanaafuu
ta'ullee
ta'uyyu
ta'uyyuu
tawullee
teenya
teessan
tiyya
too
tti
utuu
waa'ee
waan
waggaa
wajjin
warra
woo
yammuu
yemmuu
yeroo
yommii
yommuu
yoo
yookaan
yookiin
yoolinimoo
yoom

Appendix IV- Implementation Code in Python

Indexing Code

```
characters = " .,!#$%^&*();:\n\t\\\"?!{}[]<>0123456789"
def tokenize(document):
    terms = document.lower().split(' ')
    return [term.strip(characters) for term in terms]
s=open('stopword.txt','r')
stoplist=s.read()
os.chdir('corpus')
string=''
stem=[]
mystr=''
print ('Barbaacha keessan Galchaa!!')
str=raw_input()
F=open('query.txt','w')
F.write(str)
F.close()
for files in os.listdir("."):
        txt=open(files,'r')
        read=txt.read()
        string=read
        string=string.lower()#Normalization
        token=tokenize(string)
        for i in token:
            if i  not in stoplist:# Stop list removal
                stemmed=thestemmer(i)# Stemming
                stem.append(stemmed)
            else:
                    continue
        for x in stem:
            x1=x# conver list to string
            mystr=mystr+ x1 +' '
            f=open(name,'w')
            f.write(mystr)
            f.close()
def wordfreq(files):
    print files
    dict0={}
    f=open(files,'r')
    wordList=f.read()
    wordList=wordList.split()
    wordfreq = [wordList.count(p) for p in wordList]
    dictionary = dict(zip(wordList, wordfreq))
```

```
    count2 = 0
    for t in wordList:
        count2+=1
    dict0[files]=count2
    return dict0[files]
def countf():
    c=0
    for files in os.listdir("."):
            c=c+1
    return c
def tfidf():
    N=countf()
    f=open('vocabulary.txt','r')
    y=f.readlines()
    ff=open('post.txt','r')
    yy=ff.readlines()
    dic={}
    for x in y:
        z=x.split()
        df=z[1]
        term1=z[0]
        st=''
        for xx in yy:
            zz=xx.split()
            term2=zz[0]
            doc=zz[1]
            tf=zz[2]
            if term1==term2:
                doc=zz[1]
                wf=wordfreq(doc)
                tf1=float(tf)
                tf1=tf1/wf
                dfi=float(df)
                df1=N/(dfi+0.000000001)
                p=math.log(df1,2)
                idf=tf1*p
                st=doc+' '+term2
                dic[st]=idf
    return dic
fr=open('stemmed_query.txt','r')
red=fr.read()
if red=='':
    print 'your query is not significant term please try
again'
```

```
else:
    red=red.split()
    ff=open('vocabulary.txt','r')
    read=ff.read()
    read=read.split()
    def docfreq_query():
        dic={}
        dic2={}
        for word in red:
            df=0
            li=[]
            for files in os.listdir("."):
                if files.startswith('stem')or
files.startswith('voc')or files.startswith('post') or
files.startswith('query'):
                    pass
                else:
                    f=open(files,'r')
                    read1=f.read()
                    if word in read1:
                        df=df+1
            dic[word]=df
        return dic,df,word# returns df of the vocabulary word
    x,y,z=docfreq_query()
    d={}
    tf=1
    NN=countf()
    if y==0:
        print "no matching document"
    else:
        fl=float(NN)/float(y)
        idf=math.log(fl,2)
        we=0.5+0.5*tf
        w=we*idf
        d[z]=w
    weight=tfidf()
    query=d
    sim={}
    for key in query:
        for docword in weight:
            if key in docword:
                sim[docword]=query[key]*weight[docword]
    x= sim
    dicc={}
```

```
    for s,v in x.iteritems():
        qo=s.split()
        a=qo[0]
        dicc[a]=v
    d=dicc
    items = [(v, k) for k, v in d.items()]
    items.sort()
    items.reverse()              # so largest is first
    items = [(k, v) for v, k in items]
    i=1
    print 'The following documents are relevant to your
search'
    for x in items:
        print i,'.', x[0]
        i=i+1
```

Indexing

```
characters = ".,!#$%^&*();:\n\t\\\"?!{}[]<>0123456789"
def tokenize(document):
    terms = document.lower().split()
    return [term.strip(characters) for term in terms]
s=open('stopword.txt','r')
os.chdir('corpus')
stoplist=s.read()
s.close()

for files in os.listdir("."):
         stem=[]
         mystr=''
        txt=open(files,'r')
        read=txt.read()
        string=read
        token=tokenize(string)#Tokenizatione
        name='stemmed_'
        for i in token:
            if i  not in stoplist:# Stop list removal
                stemmed=thestemmer(i)# Stemming
                stem.append(stemmed)
            else:
                    continue
        for x in stem:
            x1=str(x)# conver list to string
            mystr=mystr+ x1 +' '
        f=open(name,'w')
        f.write(mystr)
        f.close()
dd=''
for file in os.listdir("."):
        f=open(file,'r')
        read=f.read()
        dd=dd+read+' '

def z():

    b=dd
    b=tokenize(b)
    b.sort()
    d=''
    for x in b:
        if x in d:
```

```
            continue
        else:
            d=d+x+' '
    f=open('voc.txt','w')
    f.write(d)
    f.close()
z()

kWF = open('voc.txt','r')
keywordFile = kWF.read()
document = keywordFile.split()
print 'doc',document
def docfreq():
    a=countf()

    dic={}
    dic2={}
    for word in document:
        df=0
        li=[]
        for files in os.listdir("."):
                f=open(files,'r')
                read=f.read()
                ls=read.split(' ')
                if word in ls:
                    df=df+1
                    print word,df
            else:
                pass
        dic[word]=df
        #print 'dic[word]=df', dic[word],'\n'
    return dic# returns df of the vocabulary word
def invert():
    dic={}
    dic2={}
    for word in document:
        df=0
        li=[]
        for files in os.listdir("."):
                f=open(files,'r')
                read=f.read()
                if word in read:
                    li.append(files)
        dic2[word]=li
```

```
    return dic2# returns idf of the vocabulary word
def countf():
    c=0
    for files in os.listdir("."):
        c=c+1
    return c


def colf():
    dict1={}

    for word in document:
        c=0
        for files in os.listdir("."):
                f=open(files,'r')
                read=f.read()
                ls=read.split()
                #print ls
                for i in ls:
                    if (word==i):
                        c=c+1
                        #print dict1
                    else:
                        continue
            else:
                pass

        dict1[word]=c
    return dict1

def voc():
    f=open('vocabulary.txt','w')
    x= docfreq()
    y=colf()
    di={}
    for a,b in x.iteritems():
        tu={}
        for k,v in y.iteritems():
            if a==k:
                b=str(b)
                v=str(v)
                f.write(a)
                f.write(' ')
```

```
                f.write(b)
                f.write(' ')
                f.write(v)
                f.write('\n')
voc()
def post():
    fi=open('post.txt','w')
    inv=invert()
    dic={}
    for k,v in inv.iteritems():
        doc=v
        #print k,v

        li=[]
        for y in doc:
            if y.startswith('post'):
                pass
            else:

                di={}
                di2={}
                f=open(y,'r')
                read=f.read()
                tf=read.count(k)
                #print k

                di[y]=tf
                fi.write(k)
                fi.write(' ')
                fi.write(y)
                fi.write(' ')
                x=str(tf)
                fi.write(x)
                a=findLocation(read,k)
                fi.write(' ')
                fi.write(a)
                fi.write('\n')
                li.append(di)
        dic[k]=li
    return dic
def findLocation(read,word):
    read1=read.split(' ')
    dic={}
    string=''
```

```
    for w in read1:
        if word not in dic:
            found=read.find(word)
            while found > -1:
                if word not in dic:
                    dic[word]=found
                    x=str(found)
                    string=string+x
                    found=read.find(word, found+1)
                else:
                    dic[word]=str(dic[word])+" "+str(found)+"
"
                    y=str(found)
                    string=string+','+y
                    found=read.find(word, found+1)
    return string
```

Declaration

This thesis is my original work and has not been submitted as a partial requirement for a degree in any University

_____

Gezehan Gutema Eggi

June 2012

The thesis has been submitted for examination with my approval as University advisor.

_____

Million Meshesha (PhD)

June 2012