ADDIS ABABA UNIVERSITY

COLLEGE OF NATURAL SCIENCE

SCHOOL OF INFORMATION SCIENCE

A Generic Approach towards All Words Amharic Word Sense Disambiguation

Dureti Siraj Bekeli

FEBRUARY, 2017

Addis Ababa, Ethiopia

ADDIS ABABA UNIVERSITY
COLLEGE OF NATURAL SCIENCE
SCHOOL OF INFORMATION SCIENCE

A Generic Approach towards All Words Amharic Word Sense Disambiguation

Dureti Siraj Bekeli

A Thesis submited to Addis Ababa University in partial fulfillment of the requirement for the Degree of Masters of Science in Information Science

February 2017

Addis Ababa, Ethiopia

ADDIS ABABA UNIVERSITY

COLLEGE OF NATURAL SCIENCE

SCHOOL OF INFORMATION SCIENCE

A Generic Approach towards All Words Amharic Word Sense Disambiguation

By:

Dureti Siraj Bekeli

Name and signiture of Members of the Examining Board

Name		Signiture	Date
_____	Chairman	_____	_____
Million Meshesha(PhD)	Advisor	_____	_____
Solomon Teferra(PhD)	Examiner	_____	_____
Dereje Teferi(PhD)	Examiner	_____	_____

Contents

List of Tables

List of Figures

List of Algorithms

List of Acronyms

BIRCH - Balanced Iterative Reducing and Clustering using Hierarchies

BNC - British National Corpus

CURE - Clustering Using REpresentatives

EM - Expectation Maximization

FDRE - Federal Democratic Republic of Ethiopia

IR - Information Retrieval

LDOCE - Longman Dictionary of Contemporary English

MRD - Machine-readable Dictionaries

MT - Machine Translation

NLP - Natural Language Processing

OALD - Oxford Advanced Learner's Dictionary of Current English

SVM - Support Vector Machine

WSD - Word Sense Disambiguation

Acknowledgment

First I would like to thank My God, because He has dealt bountifully with me. It is the Lord who make every thing possible for me and I have no words to thank Him.

My special gratitude goes to my advisor Dr. Million M. for his accommodating advise, support, understanding and patience. I have learned a lot of things from him beside acadamic staff. I am so gratful and pleased because of the chance I got to work with him. I don't have equivalent words to express my appreciation for him.

Next I would like to express thanks for my parents; my dad, Siraj Bekeli and my mom, Emebet Zewde for their Love, encouragement, advise and care. They are the reason for the person I become today. They were there, for me in hard times and engouraged me all the time. Thank you very much dad and mom.

I would like to show gratitude to my twin sister Hawi Siraj and Feruza Tujar, and my aunts Meseret Nigatu, Buni Bekeli, Sara Bekeli, Bose Bekeli and Hana W/Yohanis for their prayer and encouragement. They all are like a mother for me. They were there with me through out my thesis work. Thank you all.

I would like to thank my uncle Hassen Bekeli for his support and advise, and all my friends, for their friendship, kindness and support during my stay. And I am very thankful for Dr. Aklilu and W/r Debritu, who are Amharic Language experts. They guide and help me in studing Amharic Language.

Last but not least, I would like to thank Gender office of Addis Ababa University(AAU) for giving me this scholarship chance to study my MSC.

Abstract

Sense disambiguation is an “intermediate task” which is helpful in other NLP tasks like machine translation, information retrieval and hypertext navigation, content and thematic analysis, grammatical analysis, speech processing and text processing. This study attempts to explore a more general approach to develop a WSD for Amharic language.

To this end, a WSD system that identifies a sense of an Amharic ambiguous word by using information from tagged example sentences and Word-Net is developed. The system identifies the sense by measuring similarity between the input sentence and tagged example sentences. Two similarity measures are explored: Cosine similarity and Jaccard Coefficient similarity measure. We have collected 100 example sentences for each sense of the selected Amharic ambiguous words. The Word-Net is composed of words with their sysnonyms and gloss definition.

The performance of the system is tested using 9 nouns, 3 verbs, 3 adjectives and 2 adverbs, a total 17 words which are selected randomly. The experiments were done for disambiguating one target word in a given text.The experimental step is designed in such a way that, first the performance of Cosine similarity and Jaccard coefficient are checked individually for WSD, next Lesk algorithm is tested on the third experiment and then experiments were conducted to check the performance of the two similarity measures as combined with Lesk algorithm. The result showed that Jaccard coefficient combined with Lesk algorithm come up with the highest result, which is 89.83% accuracy. The major challenge during the disambiguation process is that for those words that are frequently collocated with similar words in their different senses the system come up with a least accuracy.

CHAPTER ONE

Introduction

1.1. Background

In all the major languages around the world, there are a lot of words which denote meanings in different contexts. A normal human being has an inborn capability to differentiate the multiple senses of an ambiguous word in a particular context, but the machines run only according to the instructions. So it is difficult for the machine to distinguish the different sense of a word based on the context. Unless and otherwise a word sense disambiguation(WSD) tool is integrated [**1**].

WSD is defined as the task of assigning the appropriate meaning (sense) to a given word in a text or discourse [**2**]. In computational linguistics, WSD is an open problem of natural language processing, which governs the process of identifying which sense of a word (i.e. meaning) is used in a sentence, when the word has multiple meanings (polysemy). WSD is considered an AI-complete problem, that is, a task whose solution is at least as hard as the most difficult problems in artificial intelligence [**3**].

The task of WSD is a historical one in the field of Natural Language Processing (NLP). In fact, it was conceived as a fundamental task of Machine Translation (MT) already in the late 1940s [**3**]. At that time, researchers had already in mind essential ingredients of WSD, such as the context in which a target word occurs, statistical information about words and senses, knowledge resources, etc [**3**]. In 1949, Zipf proposed his "Law of Meaning" theory [**1**]. This theory states that there exists a power-law relationship between the more frequent words and the less frequent words. The more frequent words have more senses than the less frequent words. The relationship has been confirmed later for the British National Corpus [**1**].

In 1950, Kaplan determined that in a particular context two words on either side of an ambiguous word are equivalent to the whole sentence of the context [**4**]. Very soon it became clear that WSD was a very difficult problem, also given the limited means available for computation. In 1957, Masterman proposed his theory of finding the actual sense of a word using the headings of the categories present in Roget's International Thesaurus [**4**].

During the 1970s the problem of WSD was attached with AI approaches aiming at language understanding [**3**]. In 1975 [**1**] Wilks developed a model on "preference semantics", where the selectional restrictions and a frame-based lexical semantics were used to find the exact sense of ambiguous words. Rieger and Small in 1979 evolved the idea of individual "word experts" [**1**]. However, generalizing the results was difficult, mainly because of the lack of large amounts of machine-readable knowledge. In this respect, work on WSD reached a turning point in 1980s with the release of large-scale lexical resources, which enabled researchers like Wilks to start using different automatic methods for knowledge extraction in 1990s [**1**].

The 1990s led to the massive employment of statistical methods and the establishment of periodic evaluation campaigns of WSD systems, up to the present days. In 1986, Lesk proposed his algorithm based on overlaps between the glosses (Dictionary definitions) of the words in a sentence [**3**]. The maximum number of overlaps represents the desired sense of the ambiguous word. In this approach the Oxford Advanced Learner's Dictionary of Current English (OALD) was used to obtain the dictionary definitions. This approach had shown the way to the other Dictionary-based WSD works. In 1991, Guthrie used the subject codes to disambiguate the exact sense using the Longman Dictionary of Contemporary English (LDOCE) [**3**].

Today, Word-Net is used as an important online sense inventory in WSD research. Statistical and machine learning methods are also successfully used in the sense classification problems. Methods that are trained on manually sense-tagged corpora (i.e., supervised learning methods) have become the mainstream approach to WSD. Corpus based WSD was first implemented by Brown in 1991 [**1**].

Sense disambiguation is an "intermediate task" which is not an end in itself, but rather is necessary at one level or another to accomplish most natural language processing tasks. It is obviously essential for language understanding applications such as message understanding, man-machine communication, etc.; it is at least helpful, and in some instances required, for applications whose aim is not language understanding such as machine translation, information retrieval and hypertext navigation, content and thematic analysis, grammatical analysis, speech processing and text processing [**4**].

WSD task has two variants: "lexical sample" and "all words" task [**3**]. The former is a system which is required to disambiguate stricted set of target words usually occurring one per sentence. The latter is deemed a more realistic form of evaluation where systems are expected to disambiguate all open-class words in a text (i.e., nouns, verbs, adjectives, and adverbs). The corpus is more expensive to produce because human annotators have to read the definitions for each word in the sequence every time they need to make a tagging judgment, rather than once for a block of instances for the same target word [**3**].

Since the 1950s, many approaches have been proposed for assigning senses to words in context, although early attempts only served as models for toy systems [**2**]. Currently, as noted by [**1**] there are two main methodological approaches in this area: knowledge-based and corpus-based methods. Knowledge-based methods use external knowledge resources, which define explicit sense distinctions for assigning the correct sense of a word in context. Corpus-based methods use machine-learning techniques to induce models of word usages from large collections of text examples [**2**].

Both knowledge-based and corpus-based methods present different benefits and drawbacks [**2**]. The major benefit of knowledge based WSD is that the systemsare proven to be ready-to-use and scalable tools because they do not require sense-annotated data. While the corpus based approach involves organizing a large number of example sentences which makes it difficult task. However, supervised corpus based algorithms have obtained better precision than knowledge-based ones. Researches on Amharic Word sense disambiguation has been conducted by various researchers using supervised, unsupervised and semi-supervised approach. In this study we will investigate application of semantic similarity measure for development of WSD system for all Amharic word classes.

1.2. Statement of the Problem

Word Sense Disambiguation remains one of the most complex problems facing computational linguists to date. Despite the advances in natural language processing (NLP), WSD is still considered one of the most challenging problems in the field [**3**]. Ever since the field's inception, WSD has been perceived as one of the central problems in NLP as an enabling technology that could potentially have far reaching impact on NLP applications in general like, machine

translation, information extraction, question answering, information retrieval, text classification, and text summarization). For this reason, many international research groups are working on WSD, using a wide range of approaches [**4**].

Amharic is the second-most spoken Semitic language in the world, after Arabic [**5**]. According to 2007 census there are more than 25 million native Amharic speakers [**6**] and the 1995 Federal Democratic Republic of Ethiopia (FDRE) constitution declare Amharic as the working language of Ethiopia. There are also speakers of Amharic language in a number of other countries, particularly Eritrea, Canada, the USA and Sweden [**7**]. Nowaday's digital resources in Amharic language are increasing from time to time on the World Wide Web(WWW). Despite all this, the language is under-resourced and Amharic language processing is still an open research area. Especially, since WSD is a central problem in NLP a lot has to be done on the area.

Some researchers have conducted research on Amharic WSD. Solomon [**8**]and Solomon [**9**]conducted study on Amharic WSD by using five selected Amharic verbs like አጠና*(eTena),* መሳል*(mesal),*መሣሣት*(me`sa`sat),*መጥራት*(metrat) and* ቀረጸ*(qereSe).* Solomon [**8**] employed supervised machine learning approach whereas Solomon [**9**] used unsupervised machine learning approach. Similarly, Getahun [**10**] conducted research based on semi supervised approach on selected words like *atena* (አጠና), *derese* (ደረሰ), *tenesa* (ተነሳ), *bela* (በላ) and *ale* (አለ). Gashaw [**11**]on the other hand selected 10 ambiguous words:Ale (አለ), Bela(በላ), Akebere(አከበረ),Atena (አጠና),Derese (ደረሰ),Melese (መለሰ), Amelekete (አመለከተ), Qetere (ቀጠረ), Tenesa (ተነሳ) and Metrat (መጥራት) for his research.

All researchers employed machine learning techniques and Getahun [**10**] has come up with the highest result using semi-supervised approach. However, there is still a gap to fill in this research area. The researchers have experimented only on a single Amharic word class which is a verb using few words. However, according to Baye [**12**] Amharic words can be classified in to five classes these are nouns, verbs, adjectives, adverbs and conjunctions. And in their study they produced a model only for those words which are selected for their experiment. The aim of this study is therefore to investigate a more generic approach to develop a WSD. For this purpose, the study has been conducted with the intention of applying similarity measures towards the development of WSD system for all Amharic word classes.

The following research questions are formulated to investigate and answer:

- What are the suitable similarity measures for all Amharic word sense disambiguation?
- How can we design and implement a WSD system prototype that will be applicable for all Amharic words?
- Which similarity measure performs best for disambiguation of Amharic ambiguous words based on a given context?

1.3. Objective of the study

1.3.1. General Objective

The general objective of this research is to explore the application of similarity measures towards the development of WSD for all-Amharic words.

1.3.2. Specific Objectives

In order to achieve the above general objective, the following specific objectives are addressed:

- To review related researches in order to understand the problem area and different methods that are used to solve the problem;
- To study Amharic language so as to understand the nature of the language and ambiguity of words;
- To prepare lexical resources that are used as a knowledge source for disambiguation;
- To prepare and preprocess a data set for training and testing;
- To explore and select the suitable semantic similarity measures;
- To design and build a prototype for Amharic WSD system prototype ;
- To conduct experiment and test the performance of the system.

1.4. Scope and limitation of the study

The aspiration of this thesis is to explore an approach to automatically disambiguate Amharic words. To this end, the design and implementation of WSD prototype is done. Eventhough, there are many approaches for WSD we have implemented only corpus based Lesk algorithm with two similarity measuring techniques: cosine similarity and Jaccard coefficient. Among the different types of word sense relationships, we have used a synonymy of the senses of the words

to identify the sense of the target word.

Because of lack of resources and difficulty of organizing example sentences for different senses of words, we have selected 17 words for the experiments. Some of the selected words have more than two senses that are given by a dictionory but we have been able to select only up to three senses. In all word WSD task, the prototype is expected to disambiguate all open-class words in a text (i.e., nouns, verbs, adjectives, and adverbs)[**3**]. So the words selected for the experiment are from this four word classes. However, the experiments are limmited to disambiguation of a single target word from a given text. This is because, tagged example sentences were collected for only 17 words.

1.5. Significance of the Study

The result of this study produces experimental evidences that demonstrate the use of corpus based Lesk algorithm for the development of Amharic WSD system. The study contribute for future researches and development in the different areas of NLP specifically in machine translation, speech processing, text processing, Information Retrieval, gammatical analysis, content and thematic analysis as those areas require WSD as complement.

In addition, this study illustrates the application of similarity measures for disambiguating word sense by using tagged example sentences. The experimental results shows the possibility of applying a WSD system that would work for all Amharic word classes which will pave a way for future researchers to investigate and study more on all-words WSD for Amharic.

1.6. Methodology of the study

Research methodology is a systematic way to solve a problem and is the procedures by which researchers go about their work of describing, explaining and predicting phenomena. The role of the methodology is to carry on the research work in a scientific and valid manner. The methodology consists of procedures and techniques for conducting a study [**14**]. Accordingly, the tools and techniques that are employed in this study to achieve the specified objectives is described below.

1.6.1. Research Design

Research design is the arrangement of conditions for collection and analysis of data in a manner that aims to combine relevance to the research purpose with economy in procedure. Research design is needed because it facilitates the smooth sailing of the various research operations, thereby making research as efficient as possible yielding maximal information with minimal expenditure, time and money [**15**].

For this study we have used empirical research method which is data-based research, coming up with conclusions which are capable of being verified by observation or experiment. In such a research it is necessary to get at fact firsthand, at their source, and actively to go about doing certain things to simulate the production of desired information. The researcher sets up experimental designs which will manipulate the persons or the materials concerned so as to bring forth the desired information [**15**].

1.6.2. Dataset preparation

To achieve the objective of the research, preparation of dataset is one important task. For this purpose, sentences containing the ambiguous words are collected from different sources on the internet. The sentences are taken from news, megazins, spritual scripts, opnions and reports from govermental and non-govermental organizations.We choose these sources to be able to capture the different senses of the selected words. As shown in table 1.1, we have selected 9 nouns, 3 verbs, 3 adjectives and 2 adverbs, a total of 17 words randomly from the Amharic-Amharic dictionary [**16**] to test the WSD prototype.

Word classes	**Words**	**Senses**		
		1	2	3
Nouns	አካል	ሰውነት,ገላ	ክፍል	ማህበር
	ዘር	ትውልድ,ዝርያ	ከፍሬ የሚገኝ	-
	እድገት	የአካል ክፍል መጨመር	የተሻለ ማግኘት	-
	ድካም	ዝለት	ጥረት	-
	መንገድ	ብልሃት, ዘዴ,አሰራር	ጎዳና	-
	ፈተና	ሙከራ,ፍተሻ, ጥያቄ	ችግር, መከራ, ስቃይ	-
	ልጅ	የአበራከ ክፋይ	ህጻን	-

	ድምፅ	ጩህት	ምርጫ, ውሳኔን ማሳወቅ	-
	ዋጋ	ፋይዳ, ጥቅም	ተመን	-
Verbs	ደረሰ	ተፈጸመ ሆነ	በቃ	-
	ቆረጠ	ገመደ	ወሰነ	-
	አከበረ	በአለ አከበረ	ከፍ አደረገ	-
Adjectives	ሃሰተኛ	ውሸት የሚናገር	ትክክለኛ ያልሆነ	-
	ደማቅ	የጎላ	የሞቀ	-
	ትኩስ	ሙቅ	አዲስ, ለጋ	-
Adverbs	በቀስታ	በዝግታ በርጋታ	በዝቅተኛ ድምፅ	-
	ወደፊት	ለሚቀጥለው ጊዜ	ከፊት አቅጣጫ	-

Table 1.1: list of words used for testing with their respective senses

The dataset is preprocessed before being used by the WSD system. The dataset will be first tokenized, in order to analyze text into a sequence of discrete tokens (words). After tokenizing the sentences, stop word removal, which is the removal of most frequently appearing words, and stemming, a task of bringing words in to their root form is done. The other preprocessing task is normalization which helps to bring texts in to standard forms.

1.6.3. Tools and Techniques

Python programming language is used for preprocessing the dataset and the development of the WSD system. The system is implemented on python 2.7.11 IDE. Python is easy to learn and powerful programming language that has efficient high-level data structures. Python has elegant syntax and dynamic typing with its interpreted nature, which makes it an ideal language for scripting and rapid application development in many areas [**17**]. This tool is chosen because of the familiarity of the researcher to the tool, it's language independency and freely available on the internet.

In this research, knowledge based approach using manually constructed Amharic Word-Net is employed for disambiguating input text. We implement corpus based Lesk algorithm which the variation of simple Lesk algorithm that is frequently used to solve the semantic ambiguity of a target word and relies on the calculation of the word overlap between the sense definitions of two or more target words and manually annotated corpora [**18**]. The similarity between the input text

and the corpus is computed by using two similarity measures: cosine similarity and Jaccard coefficient. This two similarity measures are selected because they are most popular similarity measures and can easily be implemented [**19**].

1.6.3.1. Testing and Experimental procedure

Five experiments are conducted in this thesis. First, the selected similarity measures (Jaccard coefficient and cosine similarity) and Lesk algorithm are experimented individually and then combination of similarity measures with lesk algorithm are tested for disambiguation of the ambiguous words. We have arranged dataset to test the WSD system prototype. The test dataset incorporate 10 manually tagged test sentences for each senses of the selected ambiguous words. The accuracy of the system is measured to evaluate the performance of the system.

1.7. Organization of the thesis

This thesis is organized in to six chapters. The first chapter is the introduction part of the thesis which includes the background, statement of the problem, objectives of the study, scope and limitation of the research, methodology and significance of the study. The second chapter is the literature review part. In this chapter the conceptual review of WSD such as the overview of different methodological approachs and evaluation meachanisms are presented. In addition, review of related works in the area of WSD for local languages are summerized in this chapter to show the contribution of the current study. The third chapter introduces the Amharic language which incorporates Amharic writing system, Amharic word classes and Ambiguity in Amharic language. Chapter four presents the design of the WSD system prototype which embraces the architecture of the WSD prototpe, description of preparation of Word-Net and the corpus and the similarity measures. The fifth chapter discusses the preparation of test dataset, experimental procedure, findings and challenges of the study in detail. And the last chapter, chapter six, presents conclusion and recommendations.

CHAPTER TWO

Literature Review

A word is assumed to have a finite number of discrete senses, often given by a dictionary, thesaurus, or other reference sources [**20**]. For that reason, human language is ambiguous, so that many words can be interpreted in multiple ways depending on the context in which they occur. For instance, the Amharic word ፈተና *(fetena)* can mean "Exam" or "challenge" depending on the context. Therefore, a program is needed to make forced choice between these senses of each usage of an ambiguous word based on the context [**20**]. The computational identification of meaning for words in context is called *word sense disambiguation* (*WSD*) [**3**].

According to Manning [**20**], WSD refers to a task that automatically assigns a sense, selected from a set of pre-defined word senses to an instance of a polysemuous word in a particular context. So the goal of this task is to ground the meaning of words in certain contexts into concepts as defined in some dictionary or lexical repository [**4**]. The task necessarily involves two steps [**4**]. The first step is the determination of all the different senses for every word relevant (at least) to the text or discourse under consideration. This is followed by the design of a means to assign each occurrence of a word to the appropriate sense.

Disambiguating word senses has the potential to improve many NLP tasks, such as machine translation, information retrieval, question-answering, text categorization and speech synthesis [**2**]. Although WSD is an important task it is challenging problem in the area of NLP, because of basically two reasons. First, dictionary-based word sense definitions are ambiguous. Even if trained linguists manually tag the word sense, the inter-agreement is not as high as would be expected. That is, different annotators may assign different senses to the same instance. Second, WSD involves much world knowledge or common sense, which is difficult to verbalize in dictionaries [**20**]. In the following subsections the applications, tasks, methodological approaches of WSD and evaluation mechanisms are discussed in detail.

2.1. Applications of WSD

WSD is a potential intermediate task for many other NLP systems. The main field of application of WSD is Machine Translation, but it is used in near about all kinds of linguistic researches.

Furthermore, the need of enhanced WSD capabilities appears in many applications whose aim is not language understanding. Among others, to mention some of them [**1**] [**3**] [**21**]:

- **Machine translation (MT):** This is the field in which the first attempts to perform WSD where carried out. WSD is required for machine translations, as a few words in every language have different translations based on the contexts of their use. There is no doubt that some kind of WSD is essential for the proper translation of polysemous words.
- **Information Retrieval (IR):** WSD can be used in IR in order to discard occurrences of words in documents appearing with inappropriate senses. An accurate disambiguation of the document base, together with a possible disambiguation of the query words, would allow it to eliminate documents containing the same words used with different meanings (thus increasing precision) and to retrieve documents expressing the same meaning with different wordings (thus increasing recall).
- **Information extraction (IE) and text mining:** WSD plays an important role for information extraction in different research works, where it is interesting to distinguish between specific instances of concepts, as Bioinformatics research, Named Entity recognition system, co-reference resolution etc.
- **Semantic Parsing:** WSD can be applied in restricting the space of competing parses, especially, for the dependencies, such as prepositional phrases.
- **Speech Synthesis and Recognition**: WSD could be useful for the correct phonetisation of words in Speech Synthesis, and for word segmentation and homophone discrimination in Speech Recognition.

2.2. Word Sense

The meaning of a word can vary enormously given the context. We represent some of this contextual variation by saying that the lemma *bank* has two senses. A **word sense** [**3**]is a discrete representation of one aspect of the commonly accepted meaning of a word. There are various relationships between word senses [**22**]:

- **Homonymy:** The word homonym is used for two senses which share both spelling and an orthography. A special case of multiple senses that causes problems especially for

speech recognition and spelling correction is homophones. Forexample, "ደማቅ" is homonymy having two senses ("የነላ") **vs** ("የሞቀ").

- **Homophones:** are senses that are linked to lemmas with the same pronunciation but different spelling, such as *wood*/*would* or *to*/*two*/*too*. A related problem for speech synthesis is homographs which are distinct senses linked to lemma with the orthographic form but different pronunciations. But there is no such kind of case for Amharic words.
- **Polysemy:** Sometimes, however, there is some semantic connection between the senses of a word. When two senses are related semantically, we call the relationship between them polysemy rather than homonymy. In many cases of polysemy the semantic relation between the senses is systematic and structured.
- **Metonymy:** is a subtype of Polysemy which is the use of one aspect of a concept or entity to refer to other aspects of the entity, or to the entity itself. Thus we are performing metonymy when we use the phrase *the White House* to refer to the administration whose office is in the White House.
- **Synonymy**: When the meanings of two senses of two different words (lemmas) are identical or nearly identical we say the two senses are **synonyms**. A more formal definition of synonymy that two words are synonymous if they are substitutable one for the other in any sentence without changing the truth conditions of the sentence. We often say in this case that the two words have the same propositional meaning. Words like "ዋጋ" and "ተመን" are related with this kind of sense relationship.
- **Antonyms**, by contrast, are words with opposite meaning. It is difficult to give a formal definition of antonyms. Two senses can be antonyms if they define a binary opposition or are at opposite ends of some scale. This is the case for *"ጥቁር"/"ነጭ"*, *"አጭር"/"ረጅም"*, or "ትልቅ"/*"ትንሽ"*, which are at opposite ends of the *length* or *size* scale. Another group of antonyms is **reversers**, which describe some sort of change or movement in opposite directions.
- **Hyponymy:** One sense is a **hyponym** of another sense if the first sense is more specific, denoting a subclass of the other; For example, *"እንስሳ"* is a hyponym of *"ድመት"*. Conversely, we say that *"ድመት"* is a **hyponyms** of *"እንስሳ"*. It is unfortunate that the two words (hyponyms and hyponym) are very similar and hence easily confused; for this reason the word **super ordinate** is often used instead of **hypernyms**.

2.3. Word Sense Disambiguation Tasks

There are two variants of the generic WSD tasks [**3**] [**22**]: the lexical sample task and the all-word task. In the **lexical sample** task, a small pre-selected set of target words is chosen, along with an inventory of senses for each word from some lexicon. Since the set of words and the set of senses is small, supervised machine learning approaches are often used to handle lexical sample tasks. For each word, a number of corpus instances (context sentences) can be selected and hand-labeled with the correct sense of the target word in each. Classifier systems can then be trained using these labeled examples. Unlabeled target words in context can then be labeled using such a trained classifier. Early work in word sense disambiguation focused solely on lexical sample tasks of this sort, building word-specific algorithms for disambiguating single words like *ፈተና, ዋጋ or ዘር*.

In contrast, in the **all-words** task systems are given entire texts and a lexicon with an inventory of senses for each entry, and are required to disambiguate every content word in the text. *All-words WSD*, where systems are expected to disambiguate all open-class words in a text (i.e., nouns, verbs, adjectives, and adverbs). The all-words task is very similar to part-of-speech tagging, except with a much larger set of tags, since each lemma has its own set. A consequence of this larger set of tags is a serious data sparseness problem; there is unlikely to be adequate training data for every word in the test set. On the other hand, other approaches, such as knowledge-lean systems, rely on full-coverage knowledge resources, whose availability must be assured [**3**].

2.4. Knowledge Sources for WSD

Knowledge is a fundamental component of WSD. Knowledge sources [**3**] [**4**] provide data which are essential to associate senses with words. We can see the knowledge sources by classifying them in to structured resources and unstructured resources.

2.4.1. Structured Resources

According to Robert [**3**] and Nancy [**4**], structured resources include the following:

- **Thesauri**

Thesaurus is structured knowledge source which provide information about relationships between words, like synonymy, antonyms and, possibly, further relations. The most widely used

thesaurus in the field of WSD is Roget's International Thesaurus which was put into machine-tractable form in the 1950's and has been used in a variety of applications including machine translation, information retrieval, and content analysis. The thesaurus also supplies an explicit concept hierarchy consisting of up to eight increasingly refined levels. Typically, each occurrence of the same word under different categories of the thesaurus represents different senses of that word; i.e., the categories correspond roughly to word senses. A set of words in the same category are semantically related. The latest edition of the thesaurus contains 250,000 English word entries organized in six classes and about 1000 categories. Some researchers also use the Macquarie Thesaurus by Bernard in 1986, which encodes more than 200,000 synonyms.

- ***Machine-readable dictionaries*** **(MRDs)**

MRD have become a popular source of knowledge for natural language processing since the 1980s, when the first dictionaries were made available in electronic format: Collins English Dictionary, the Oxford Advanced Learner's Dictionary of Current English, the Oxford Dictionary of English, and the Longman Dictionary of Contemporary English (LDOCE). A primary area of activity during the 1980's involved attempts to automatically extract lexical and semantic knowledge bases from MRDs. While dictionaries provide detailed information at the lexical level, they lack pragmatic information that enters into sense determination.

- **Ontologies**

Ontologies are specifications of conceptualizations of specific domains of interest, usually including taxonomy and a set of semantic relations. In this respect, Word-Net and its extensions can be considered as ontologies. Word-Net is a computational lexicon of English based on psycholinguistic principles, created and maintained at Princeton University. It encodes concepts in terms of sets of synonyms (called *synsets*). Its latest version, Word-Net 3.0, contains about 155,000 words organized in over 117,000 synset. There are various Word-Nets constructed for Languages other than English. Table 2.1 presents list of some Word-Nets in different languages [**23**].

Language	Resource name	Developer(s)
English	EuroWord-Net English	University of Sheffield
French	WoNeF	CEA-LIST
French	WOLF	ALPAGE team: INRIA, Universite Paris Diderot – Paris 7
German	GermaNet	Universitat Tubingen
Hindi	Hindi Word-Net	Indian institute of Technology Bombay Powai, Mumbai
Arabic	Arabic Word-Net	Arabic Word-Net
Multilingual(200+ languages)	Open Multilingual Word-Net	Linguistic and Multiligual Studies, NTU

Table 2.1: Some Word-Nets in the world

2.4.2. Unstructured resources

The unstructured knowledge resource is a corpora, which is collections of texts used for learning language models. Corpora can be sense-annotated or raw (i.e., unlabeled). Both kinds of resources are used in WSD, and are most useful in supervised and unsupervised approaches, respectively. The most popular Raw corpora are the following: one is the Brown Corpus, a million word balanced collection of texts published in the United States in 1961; the other is the British National Corpus (BNC), a 100 million word collection of written and spoken samples of the English language (often used to collect word frequencies and identify grammatical relations between words). There is also the Wall Street Journal (WSJ) corpus, a collection of approximately 30 million words from WSJ; the American National Corpus, which includes 22 million words of written and spoken American English; etc. Sense-Annotated Corpora are corpora like SemCor, the largest and most used sense-tagged corpus, which includes 352 texts tagged with around 234,000 sense annotations; MultiSemCor, an English-Italian parallel corpus annotated with senses from the English and Italian versions of Word-Net; the line-hard-serve corpus containing 4000 sense-tagged examples of these three words (noun, adjective, and verb, respectively); the interest corpus with 2369 sense-labeled examples of noun interest to mention some of them [**3**].

2.5. Approaches in Word Sense Disambiguation

Since the 1950s, many approaches have been proposed for assigning senses to words in context, although early attempts only served as models for toy systems. Currently, there are two main methodological approaches in this area: knowledge-based and corpus-based methods [**2**].

2.5.1. Knowledge based approach

Knowledge-based methods use external knowledge resources, which define explicit sense distinctions for assigning the correct sense of a word in context. The aim of Knowledge based approach (Dictionary based approach) WSD is to exploit knowledge resources to infer the senses of words in context. Work on WSD reached a turning point in the 1980s and 1990s when large-scale lexical resources such as dictionaries, thesauri, and corpora became widely available. These approaches, by exploiting the knowledge contained in the dictionaries, mainly seek to avoid the need for large amounts of training material. Word-Net is the mostly used machine readable dictionaries in this research field [**2**] [**24**]. There are various algorithms to knowledge based WSD, three of them are discussed below:

2.5.1.1. The Lesk Algorithm

Lesk algorithm [**1**]is the first machine readable dictionary based algorithm built for word sense disambiguation. This algorithm depends on the overlap of the dictionary definitions of the words in a sentence. In this approach, First of all a short phrase (containing an ambiguous word) is selected from the sentence. Then, dictionary definitions (glosses) for the different senses of the ambiguous word and the other meaningful words present in the phrase are collected from an online Dictionary. Next, all the glosses of the key word are compared with the glosses of other words. The sense, for which the maximum number of overlaps occurs, represents the desired sense of the ambiguous word. The simplest version of the algorithm, often called the **Simplified Lesk** algorithm by Kilgarriff and Rosenzweig presented in algorithm 2.1. below: [**18**]

(1) for each sense *i* of *W*1

(2) for each sense *j* of *W*2

(3) compute *Overlap(i,j)*, the number of words in common between the definitions of sense *i* and sense *j*

(4) find *i* and *j* for which *Overlap(i,j)* is maximized

(5) assign sense *i* to *W*1 and sense *j* to *W*2

Algorithm 2.1: Simplified Lesk algorithm

The primary problem [**25**] with either the original or simplified approaches, however, is that the dictionary entries for the target words are short, and may not provide enough chance of overlap with the context. There are Variations of the Lesk Algorithm such as simulated annealing and Augmented Semantic spaces [**18**].

- **Simulated Annealing**

The major problem with the original Lesk algorithm is it leads to a combinatorial explosion when applied to the disambiguation of more than two words. As cited on [**18**] Cowie et al.(1992) proposed simulated annealing as a solution to this problem. Cowie define a function *E* that reflects the combination of word senses in a given text, and whose minimum should correspond to the correct choice of word senses. For a given combination of senses, all corresponding definitions from a dictionary are collected, and each word appearing at least once in these definitions receives a score equal to its number of occurrences. Adding all these scores together gives the redundancy of the text. The *E* function is then defined as the inverse of redundancy, and the goal is to find a combination of senses that minimizes this function. To this end, an initial combination of senses is determined (e.g., pick the most frequent sense for each word), and then several iterations are performed, where the sense of a random word in the text is replaced with a different sense, and the new selection is considered as correct only if it reduces the value of the *E* function. The iterations stop when there is no change in the configuration of senses [**18**]. Algorithm 2.2 shows how simulated annealing works [**18**]:

(1) for each sense *i* of *W*

(2) determine *Overlap(i)*, the number of words in common between the definition of sense *i* and current sentential context

(3) find sense *i* for which *Overlap(i)* is maximized

(4) assign sense *i* to *W*

Algorithm 2.2: Algorithm for Simulated Annealing

- **Corpus based Lesk algorithm**

A corpus based Lesk algorithm [**18**]is other variation of the Lesk algorithm which is frequently used to solve the semantic ambiguity of a target word, using manually annotated corpora. This corpus-based variation has the capability to augment the sense-centered context of a word with additional tagged examples. Subsequently, the most likely sense for a new occurrence of the ambiguous target word is identified as the one with the highest dictionary overlap between the sense-centered contexts and the new context. The steps of Corpus based algorithm are presented below in algorithm 2.3: [**18**]

(1) for each sense *i* of *W*

(2) set *Weight(i)* to 0

(3) for each [unique] word *w* in surrounding context of *W*

(4) if *w* appears in the training examples or Dictionary definition of sense *i*

(5) add *Weight(w)* to *Weight(i)*

(6) choose sense *i* with highest *Weight(i)*

Algorithm 2.3: Corpus based Lesk algorithm

The weight of a word is defined using a measure borrowed from the information retrieval community: Weight (w) is the inverse document frequency (IDF) of the word *w* over the examples and dictionary definitions. The IDF of a word is –log(*p*(*w*)), where *p*(*w*) is estimated as the fraction of examples and definitions including the word *w* [**18**].

- **Augmented Semantic Spaces**

Another variation of the Lesk algorithm, called the adapted Lesk algorithm [**18**], which was introduced by Banerjee and Pedersen. In this algorithm the definitions of related words are used in addition to the definitions of the word itself to determine the most likely sense for a word in a given context. Banerjee and Pedersen employ a function similar to the one defined by Cowie to determine a score for each possible combination of senses in a text, and attempt to identify the sense configuration that leads to the highest score.

While the original Lesk algorithm considers strictly the definition of a word meaning as a source of contextual information for a given sense, this algorithm is based on the Word-Net hierarchy. The algorithm takes into account hypernyms, hyponyms, holonyms, meronyms, troponyms, attribute relations, and their associated definitions to build an enlarged context for a given word meaning. Algorithm 2.4 desctibes steps followed by augmented semantic space: [**18**]

1. Select a context: optimizes computational time so if N is long, we will define K context around the target word (or k-nearest neighbor) as the sequence of words starting K words to the left of the target word and ending K words to the right. This will reduce the computational space that decreases the processing time. For example: If k is four, there will be two words to the left of the target word and two words to the right.

2. For each word in the selected context, we look up and list all the possible senses of both POS (part of speech) noun and verb.

3. For each sense of a word (Word Sense), we list the following relations (example of pine and cone):

   - Its own gloss/definition that includes example texts that Word-Net provides to the glosses.
   - The gloss of the synsets that are connected to it through the hypernym relations. If there is more than one hypernym for a word sense, then the glosses for each hypernym are concatenated into a single gloss string (*).
   - The gloss of the synsets that are connected to it through the hyponym relations (*).
   - The gloss of the synsets that are connected to it through the meronym relations (*).
   - The gloss of the synsets that are connected to it through the troponym relations (*).

(*) All of them are applied with the same rule.

4. Combine all possible gloss pairs that are archived in the previous steps and compute the relatedness by

searching for overlap. The overall score is the sum of the scores for each relation pair.

Algorithm 2.4: Augmented Semantic Spaces algorithm

2.5.1.2. Selectional Restrictions and Selectional Preferences

A historical type of knowledge-based algorithm is one which exploits selectional preferences to restrict the number of meanings of a target word occurring in context.Selectional preferences or restrictions are constraints on the semantic type that a word sense imposes on the words with which it combines in sentences (usually through grammatical relationships) [**3**]. Selectional preferences capture information about the possible relations between word categories, and represent commonsense knowledge about classes of concepts. Selectional restrictions are associated with senses, not entire lexemes [**25**].

Selectional preferences are difficult to put them into practice to solve the problem of WSD becuase of the circular relation between selectional preferences and WSD: learning accurate semantic constraints requires knowledge of the word senses involved in a candidate relation, and, vice versa, WSD can improve if large collections of selectional preferences are available [**18**].The most frequently used approaches that try to overcome this circularity and automatically learn selectional preferences are based on frequency counts, information-theory measures, or class-to-class relations acquired from manually-crafted taxonomies [**18**].

Frequency counts of word-to-word relations are useful measures to account for the *semantic fit* between words. Given a pair of words w1 and w2 and a syntactic relation R (e.g., subject-verb, verb-object, etc.), this method counts the number of instances (R, w1, w2) in a corpus of parsed text, obtaining a figure Count(R, w1, w2). Another estimation of the semantic appropriateness of a word-to-word relation is the conditional probability of word w1 given the other word w2 and the relation R: [**3**]

$$P(w1 \mid w2, R) = \text{Count}(w1,w2,R)\ \text{Count}(w2,R)$$

- **Heuristic Method**

The heuristics method predicts word meanings based on heuristics drawn from linguistic properties observed on large texts. Three types of heuristics are used as a baseline for estimating WSD system: Most Frequent Sense, One Sense per Discourse and One Sense per Collocation

[**18**]. The Most Frequent Sense is very simple method which is often used as a baseline for WSD. This apprachworks by finding all likely senses that a word can have and it is basically right that one sense occurs often than the others. The drawback associated with this method is that sense distributions may not always be available, and therefore the most-frequent-sense heuristic is applicable only to those few languages for which significantly large sense-tagged corpora are available. Moreover, a change in domain can significantly affect the sense distributions, considerably decreasing the performance of this simple heuristic [**18**].

One Sense per Discourse was introduced by Gale as cited on [**1**], says that a word will preserve its meaning among all its occurrences in a given text.This is a rather strong rule since it allows for the automatic disambiguation of all instances of a certain word, given that its meaning is identified in at least one such occurrence. And finally, One Sense per Collocation is same as One Sense per Discourse except it is assumed that words that are nearer provide strong and consistent signals to the sense of a word [**1**]. It was introduced by Yarowsky in 1993, and it states that a word tends to preserve its meaning when used in the same collocation. In other words, nearby words provide strong and consistent clues to the sense of a target word. It was also observed that this effect is stronger for adjacent collocations, and becomes weaker as the distance between words increases [**18**].

2.5.2. Corpus based approach

Corpus-based approaches are those that build a classification model from examples. These methods involve two phases: *learning* and *classification* [**21**]. The learning phase consists of learning a sense classification model from the training examples. The classification process consists of the application of this model to new examples in order to assign the output senses. Most of the algorithms and techniques to build models from examples come from the Machine Learning area of AI, such as supervised and unsupervised approach [**21**].

2.5.2.1. Supervised Approach

In the last 15 years, the NLP community has witnessed a significant shift from the use of manually crafted systems to the employment of automated classification methods. Supervised WSD is an automated classification method that uses machine-learning techniques for inducing a classifier from manually sense-annotated data sets. In supervised disambiguation, a

disambiguated corpus is available for training [**26**]. Usually, the classifier (often called *word expert*) is concerned with a single word and performs a classification task in order to assign the appropriate sense to each instance of that word. The training set used to learn the classifier typically contains a set of examples in which a given target word is manually tagged with a sense from the sense inventory of a reference dictionary. In most cases, supervised approaches to WSD have obtained better results than unsupervised methods [**3**]. There are different supervised algorithms that are used for the task of WSD, the most common one's are discussed below:

A. Naïve Bayes

A Naïve Bayes classifier [**27**] is a simple probabilistic classifier based on the application of Bayes' theorem. It has been used in its most classical setting, that is, assuming the independence of features, it classifies a new example by assigning the class that maximizes the conditional probability of the class given the observed sequence of features of that example. Model probabilities are estimated during the training process using relative frequencies. To avoid the effect of zero counts, a very simple smoothing technique has been used. Despite its simplicity, Naïve Bayes is claimed to obtain state-of-the-art accuracy on supervised WSD in many articles [**27**].

The **Naïve Bayes classifier** [**25**] approach to WSD is based on the premise that choosing the best sense *s* out of the set of possible senses *S* for a feature vector amounts to choosing the most probable sense given that vector. It relies on the calculation of the conditional probability of each sense S_i of a word w given the features f_j in the context. The maximum value evaluated from the formula represents the most appropriate sense in the context [**1**]:

$$\hat{s} = \underset{s_i \in Sense_D(w)}{argmax} P(S_i|f_i, \dots, f_m) = \underset{s_i \in Sense_D(w)}{argmax} \frac{P(f_1, \dots., f_m|S_i)P(S_i)}{P(f_1, \dots., f_m)}$$

$$= \underset{s_i \in Sense_D(w)}{argmax} P(S_i) \prod_{j=1}^{m} P(f_j|S_i) \quad \dots\dots\dots\dots\dots\dots (2.1)$$

Where, the number of features is represented by m. The probability $P(S_i)$ is calculated from the co-occurrence frequency in training set of sense and $P(f_j | S_i)$ is calculated from the feature in the presence of the sense. The steps for Naïve Bayes algorithm is shown in algorithm 2.5 [**28**]:

1. Initialize context c, sense s, and ambiguous word w.
2. As per training context
3. $p(s \mid w,c)=\underset{s_i \in Sense_D(w)}{argmax} P(S_i) \prod_{j=1}^{m} P(f_j \mid S_i)$

   Calculate Maximized $p(s \mid w,c)$
4. Select one with highest value
5. Map sense according to the highest accuracy.

Algorithm 2.5: Naïve Bayes algorithm

B. Decision list

Decision list **classifiers** are equivalent to simple case statements in most programming languages. Decision list is an ordered set of "if-then-else" rules for categorizing test instance. In this setting, a Decision List is a list of features extracted from the training examples and sorted by a log-likelihood measure [**27**]. Using those rules few parameters like feature-value and sense score are created. Based on the decreasing scores, final order of rules is generated, which creates the decision list [**1**]. In this approach, a sequence of tests is applied to each target word feature vector. Each test is indicative of a particular sense [**25**].

When testing, the decision list is checked in order and the feature with the highest weight that matches the test example is used to select the winning word sense. When any word is considered, first its occurrence is calculated and its representation in terms of feature vector is used to create the decision list, from where the score is calculated. The maximum score for a vector represents the sense [**27**]. Thus, only the single most reliable piece of evidence is used to perform disambiguation. If a test succeeds, then the sense associated with that test is returned. If the test fails, then the next test in the sequence is applied. This continues until the end of the list, where default tests simply return the majority test [**25**]. Algorithm 2.6 presents the steps followed by dicision list:

1. Read data set and calculation POS
2. Prepare context containing various senses of word
3. Calculate frequency at context (i.e. - p- and +P+), (where P- Negative and P+ Positive)
4. Calculate information gain for calculating entropy (S) =-P+log2P+-P-log2P-
5. Gain (S,A)= Entropy(S) -$\sum_{v\in DA}\frac{|S_v|}{|S|}\ Entropy\ (S_v)$
6. Select highest (Entropy, Attribute ratio)

Algorithm 2.6: Decision list algorithm

C. Neural Networks

In the Neural Network [**1**] [**2**] based computational model, artificial neurons are used for data processing using connectionist approach. Input of this learning program is the pairs of input features, and goal is to partition the training context into non-overlapping sets. Next, a larger activation of these newly formed pairs is produced and link weights are gradually adjusted. Neural networks can be used to represent words as nodes and these words will activate the ideas to which they are semantically related. The aim is to use the input features to partition the training contexts into non-overlapping sets corresponding to the desired responses. As new pairs are provided, link weights are progressively adjusted so that the output unit representing the desired response has a larger activation than any other output unit. Neural networks are trained until the output of the unit corresponding to the desired response is greater than the output of any other unit for every training example. For testing, the classification determined by the network is given by the unit with the largest output. Weights in the network can be either positive or negative, thus enabling the accumulation of evidence in favor or against a sense choice.

D. Support Vector Machine

The SVM [**3**] [**21**] based algorithms are used to classify few examples into two distinct classes. The algorithms are based on the Structural Risk Minimization principle from the Statistical Learning Theory. This method is based on the idea of learning a linear hyperplane from the training set that separates positive examples from negative examples. The hyperplane is located in that point of the hyperspace which maximizes the distance to the closest positive and negative examples. The positive and negative examples which are closest to the hyperplane are called support vector [**3**]. Figure 2.1 illustrates the geometric intuition: the line in bold the plane which

separates the two classes of examples, whereas the two dotted lines denote the plane tangential to the closest positive and negative examples. [**3**]

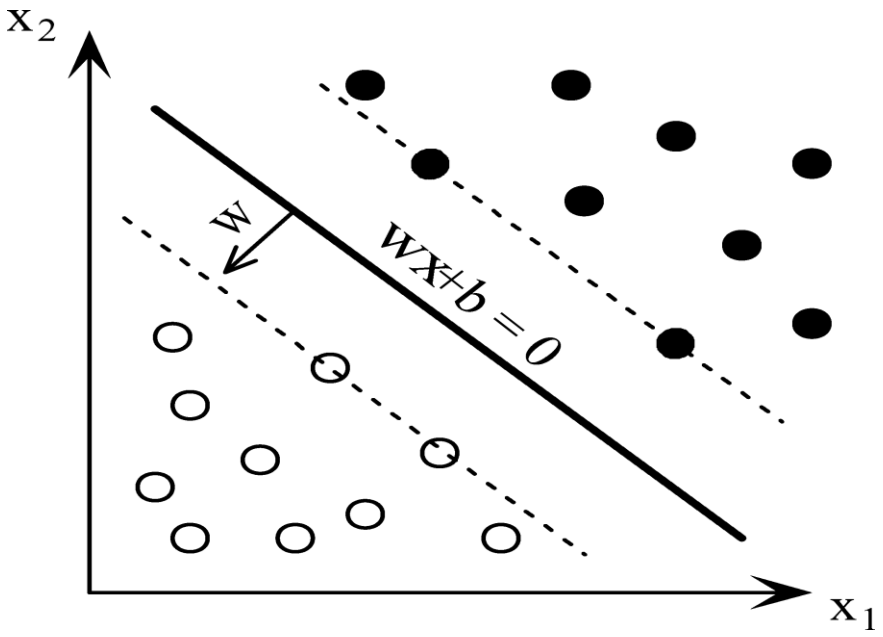


Figure 2.1: The geometric intuition of SVM

The classification of the test example depends on the side of the hyperplane, where the test example lies in. The input features can be mapped into a high dimensional space also, but in that case, to reduce the computational cost of the training and the testing procedure in high dimensional space, some kernel functions are used. A regularization parameter is used in case of non-separable training examples. The default value of this parameter is considered as 1. This regularization procedure controls the trade-off between the large margin and the low training error [**1**]. As SVM is a binary classifier, in order to be usable for WSD it must be adapted to multiclass classification (i.e., the senses of a target word). A simple possibility, for instance, is to reduce the multiclass classification problem to a number of binary classifications of the kind sense *Si* versus all other senses. As a result, the sense with the highest confidence is selected [**3**].

2.5.2.2. Unsupervised approach

Unsupervised approaches to WSD are based on the idea that the same sense of a word will have similar neighboring words. They are able to induce word senses from input text by clustering

word occurrences, and then classifying new occurrences into the induced clusters [**24**]. The unsupervised approach needs less computing time and memory space. It is suitable for online machine translation and information retrieval. This approach has two types of distributional approaches; the first one is monolingual corpora and the other is translation equivalence based on parallel corpora [**1**].

These algorithms generally do not assign meaning to the words instead they discriminate the word meanings based on information, found in un-annotated corpora. By sense discrimination we mean that one can cluster the contexts of an ambiguous word into a number of groups and discriminate between these groups without labeling them [**26**]. Admittedly, unsupervised WSD approaches have a different aim than supervised and knowledge-based methods, that is, identifying sense clusters compared to that of assigning sense labels [**24**].

Unsupervised methods have the potential to overcome the knowledge acquisition bottleneck, that is, the lack of large-scale resources manually annotated with word senses [**3**]. They do not rely on labeled training text and, in their purest version, do not make use of any machine-readable resources like dictionaries, thesauri, ontologies, etc [**24**]. However, it theoretically has worse performance than the supervised approach because it relies on less knowledge [**29**]. There are different unsupervised approaches that can be applied for construction WSD.

A. Single link and complete link clustering

In single link clustering [**20**]the similarity between two clusters is the similarity of the two closest objects that are from the two different clusters and select the pair with the greatest similarity. We search over all pairs of objects that are from the two different clusters and select the pair with the greatest similarity. Single-link clustering has clusters with the good local coherence since similarity function is locally defined. However, clustering can be elongated or "straggly "which sometimes called chaining effect. Single-link clustering is closely related to the minimum spanning tree of a set of points. A single-link clustering can be constructed top-down from an minimum spanning tree by removing the longest edge in the minimum spanning tree so that two unconnected components are created, corresponding to two sub clusters. The same operation is then recursively applied to these two sub clusters.

Complete-link clustering [**20**] has a similarity function that focuses on global cluster quality as opposed to locally coherent clusters as in that if single-link clustering. The similarity of two

clusters is the similarity of their two most dissimilar members. Complete-link clustering avoids elongated clusters. The disadvantage of complete-link clustering is that it has time complexity (n^3) since there are n merging steps and each step requires $O(n^2)$ comparisons to find the smallest similarity between any two objects for each cluster pair(where n is the number of objects to be clustered). Single-link and complete-link clustering can be graph-theoretically interpreted as finding a maximally connected and maximally complete graph.

B. K-means clustering

K-means [**25**] [**18**] is a clustering algorithm that defines clusters by the center of mass of their members. The aim of K-means algorithm is to partition *n* observations into k clusters in which each statement belongs to the cluster with the nearest mean. This results into a splitting of the data space into Verona cells. The procedure follows a simple way to cluster a given data set through a certain number of clusters (assume k clusters) fixed a priority. We need a set of initial cluster centers at the beginning. Then we go through several iterations of assigning each object to the cluster whose center is closest. After all objects have been assigned, we recomputed the center of each of cluster as the centroid or mean of its member as shown in algorithm 2.7, that is c_j. The distance function may be Euclidean distance or manhatten distance. The time complexity of K-means clustering is $O(n)$ since both steps of iteration are O(n) and only a constant number of iterations is computed.

1. Given: a set X= $\{_1,\ldots,_n\}$ R^m
2. a distance measure d: R^m R^m R
3. a function for computing the mean : P(R) R^m
4. Select k initial centers $_1,\ldots,_k$
5. **while** stopping criterion is not true **do**
6. **for** all clusters c_j do
7. $c_j = \{_1 | _l \, d(_I, _l)\}$
8. **end**
9. **for** all means $_j$ do
10. $_j = (c_j)$
11. **end**
12. **End**

Algorithm 2.7: The K-means clustering algorithm

C. Expectation Maximization (EM)

One way to introduce the EM algorithm is as a 'soft' version of K-mean clustering [**20**]. An expectation– maximization (EM) algorithm is an iterative method for finding maximum likelihood or maximum a posteriori (MAP) estimates of parameters in statistical models, where the model depends on unobserved hidden variables Θ of a probability distribution, given measurement data U. EM is usually described as two steps the Expectation step and the Maximization step. The Expectation step computes the expectation of the log likelihood evaluated using the current estimate for the parameters, and the Maximization step computes parameters maximizing the expected log-likelihood found on the *Expectation* step [**30**]. The two EM steps can also be described in six steps as shown in algorithm 2.8: [**30**]

Step 1: pick an initial guess $\theta^{(m=0)}$ for θ.

Step 2: Given the observed data y and pretending for the moment that your current guess is correct, calculate how likely it is that the complete data is exactly x, that is, calculate the conditional distribution.

Step 3: Throw away your guess, but keep Step 2's guess of the probability of the complete data.

Step 4: In Step 5 we will make a new guess of that maximizes (the expected). We'll have to maximize the expected because we don't know, but luckly in step 2 we made a guess of the probability distribution of. So, we will integrate over all possible values of, and for each possible value of, we weight by the probability of seeing that. However, we don't really know the probability of seeing that we made in step 2, which was. The expected is called the Q- function:

$$Q\left(\theta\middle|\theta^{(m)}\right) = expected\ \log p(x|\theta) = E_{(X|y),\theta^{(m)}}[\log(X|\theta)]$$

$$= \int_{x(y)} \log p(x|\theta)\, p\left(x\middle|y,\theta^{(m)}\right)dx,$$

Where you integrate over the support of x given y, x(y), which is the closure of the set {. Note that is a free variable in the above equation, so the Q-function of, and also depends on your old guess.

Step 5: make a new guess that maximizes the expected log-likelihood given in the above equation.

Step 6: let m=m+1 and go back to step 2.

Algorithm 2.8: K-means algorithm

The key property of EM algorithm [**20**] is monotonicity: with iteration of E and M steps, the likelihood of the model given the data increases. This guarantees that each iterations produces model parameters that are more likely given the data we see. However, while the algorithm will eventually move to a local maximum, it will often not find the globally best solution. EM is useful for several reasons: conceptual simplicity, ease of implementation, and the fact that each iteration improves expectation. The rate of convergence on the first few steps is typically quite good, but can become excruciatingly slow as you approach local optima. Generally, EM works best when the fraction of missing information is small and the dimensionality of the data is not too large. EM can require much iteration, and higher dimensionality can dramatically slow down the E-step.

D. BIRCH algorithms

Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) [**31**] [**32**] incrementally and dynamically clusters incoming multi-dimensional metric data points to try to produce the best quality clustering with the available resources (i. e., available memory and time constraints). BIRCH can typically find a good clustering with a single scan of the data, and improve the quality further with a few additional scans. BIRCH is also the first clustering algorithm proposed in the database area to handle "noise". It introduces two concepts, clustering feature and clustering feature tree (CF tree), which are used to summarize cluster representations. A CF tree is a height-balanced tree that stores the clustering features for a hierarchical clustering. CF tree consists a branching factor (maximum number of children per none leaf node) B and threshold T. Each internal node contains a CF triple for each of its children. Each leaf node also represents a cluster made up of all sub clusters represented by its entries and contains a CF entry for each sub cluster in it. A sub cluster in a leaf node must have a diameter no greater than a given threshold value (maximum diameter of sub-clusters in the leaf node).

An object is inserted to the closest leaf entry (sub cluster). All the entries in a leaf node must satisfy the threshold requirements with respect to the threshold value T, that is, the diameter of the sub cluster must be less than T. If the diameter of the sub cluster stored in the leaf node after insertion is larger than the threshold value, then the leaf node and other nodes are split. After the insertion of the new object, information about it is passed toward the root of the tree. The size of

the CF tree can be changed by modifying the threshold. These structures help the clustering method achieve good speed and scalability in large databases. BIRCH is also effective for incremental and dynamic clustering of incoming objects. This algorithm can find approximate solution to combinatorial problems with very large data sets [**31**].The figure 2.1 below represents an overview of BIRCH algorithm [**31**].

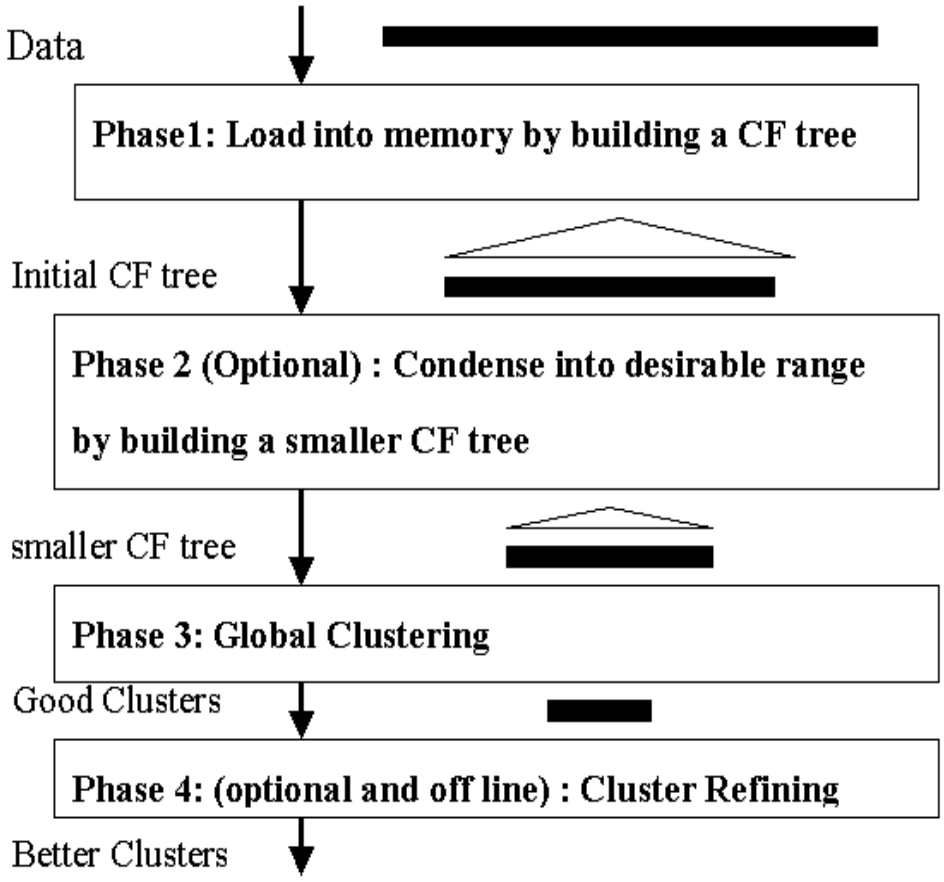


Figure 2.2: Overview of BIRCH algorithm

E. CURE Clustering algorithm

Clustering Using REpresentative(CURE) [**32**] [**18**] is improved clustering algorithm which employs a novel hierarchical clustering algorithm by adopting a middle ground between the centroid-based and the all-point extremes. Instead of using a single centroid or object to represent a cluster, a fixed number of representative points are chosen. The representative points of a cluster are generated by first selecting well-scattered objects for the cluster and then "shrinking" or moving them toward the cluster center by a specified fraction, or shrinking factor. At each step of the algorithm, the two clusters with closest pair of representative point are chosen.

Having more than one representative point per cluster allows CURE to adjust well to the geometry of non-spherical shapes. The shrinking or condensing of clusters helps dampen the effects of outliers. Therefore, CURE is more robust to outliers and identifies clusters having non spherical shapes and wide variance in size. That's why it scales well for large databases without sacrificing clustering quality.

First, instead of pre-clustering with all the data points, CURE begins by drawing a random sample from the database. Second, in order to further speed up clustering, CURE first partitions the random sample and partially clusters the data points in each partition. After eliminating outliers, the pre-clustered detail each partition is then clustered in a final pass to generate the final clusters. Once clustering of the random sample is completed, instead of a single centroid, multiple representative points from each cluster are used to label the remainder of the data set [**18**].

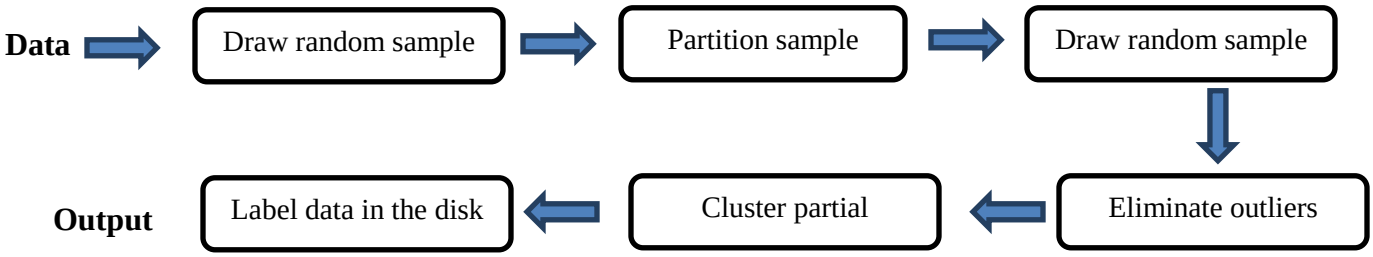


Figure 2.3: Overview of CURE algorithm

2.6. Evaluation of WSD system

Comparing and evaluating different WSD systems is extremely difficult, because of the different test sets, sense inventories, and knowledge resources adopted [**3**].The assessment of word sense disambiguation systems is usually performed in terms of evaluation measures borrowed from the field of information retrieval. The measurement methodologies are Precision, Recall and F-measure. Precision determines how good the answers given by the system are being assessed. The *precision P* of a system is computed as the percentage of correct answers given out of the total answer probided by the WSD system, that is:

$$P = \frac{correct\ answers\ provided}{total\ answers\ \ provided} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(2.2)$$

On the other hand the recall, R is defined as the number of correct answers given by the automatic system over the total number of answers to be given:

$$\mathrm{R} = \frac{\#\ \text{correct answers provided}}{\#\ \text{total answers to provide}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(2.3)$$

Where we indicate with ϵ the case in which the system does not provide an answer for a specific word w_i. The total number of answers is given by $n=|T|$. Finally, a measure which determines the weighted harmonic mean of precision and recall, called the *F*1-*measure* or *balanced F-score*, is defined as

$$F_1 = \frac{2PR}{P+R} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(2.4)$$

It has been argued that the above measures do not reflect the ability of systems to output a degree of confidence for a given sense choice. In this regard different evaluation metrics were proposed, however, precision, recall and F-measures are mostly used evaluation methodologies [**3**].

2.7. THE SENSEVAL/SEMEVAL COMPETITIONS

Senseval (now renamed *Semeval*) [**3**] is an international word sense disambiguation competition, held every three years since 1998. The objective of the competition is to perform a comparative evaluation of WSD systems in several kinds of tasks, including all-words and lexical sample WSD for different languages, and, more recently, new tasks such as semantic role labeling, gloss WSD, lexical substitution, etc. The systems submitted for evaluation to these competitions usually integrate different techniques and often combine supervised and knowledge-based methods (especially for avoiding bad performance in lack of training examples). The Senseval workshops are the best reference to study the recent trends of WSD and the future research directions in the field. Moreover, they lead to the periodic release of data sets of high value for the research community.

The first edition of Senseval took place in 1998 at Herstmonceux Castle, Sussex. Senseval-1 consisted of a lexical-sample task for three languages: English, French, and Italian. A total of 25 systems from 23 research groups participated in the competition. The English test set contained 8400 instances of 35 target words. The best systems performed with between 74% and 78% accuracy. The baseline, based on the most frequent sense, achieved a 57% accuracy. Decision lists with the addition of some hierarchical structure were the most successful approach in the first edition of the Senseval competition [**3**]. Senseval-1 produced a set of benchmarks for WSD

system performance. It set out to establish the viability of WSD as a separately evaluable NLP task [**33**].

Senseval-2 [**3**] [**33**] took place in Toulouse (France) in 2001-2002 and its goals were to encourage tasks in new languages and to broaden the range of tasks. Two main tasks: all-words and lexical sample WSD were organized in 12 different languages (such as Czech, Basque, Dutch, English, Estonian, Italian, Japanese, Korean, Spanish and Swedish) and translation task on Japanese language only. Overall, 93 systems from 34 research groups participated in the competition. The Word-Net 1.7 sense inventory was adopted for English. The best lexical sample task systems performed with between 39% and 78% accuracy and the best all-word task systems performed between 67% and 94%. The Czech language resuled the hight performance in Senseval 2. On the other hand, the best translation task systems for Japanese language have scored 79% accuracy. The performance was generally lower than in Senseval-1, probably due to the fine granularity of the adopted sense inventory. Supervised systems outperformed unsupervised approaches.

The third edition of the Senseval competition took place in Barcelona in 2004 [**3**]. It consisted of 14 tasks, and, overall, 160 systems from 55 teams participated in the tasks. These included lexical sample and all-words tasks for seven languages as well as new tasks such as gloss disambiguation, semantic role labeling, multilingual annotations, logic forms, and the acquisition of sub-categorizations. Most of the systems were supervised. The performance of the best 14 lexical sample task systems ranged between 72.9% and 70.9%, suggesting that this task seems to have reached a ceiling which is difficult to overcome. Most of the top systems used kernel methods. Other approaches include the voted combination of algorithms and the use of a rich set of features, comprising domain information and syntactic relations. The English all-words task saw the participation of 26 systems from 16 teams. The test set included 2037 sense-tagged words. The best system attained a 65.1% accuracy, whereas the first sense baseline achieved 60.9% or 62.4% depending on the treatment of multi-words and hyphenated words.

2.8. Review of Related Works

2.8.1. Local Researches on Word Sense Disambiguation

Solomon [**8**]**,** applied a corpus based WSD so as to acquire disambiguation information automatically. This study reported experiments on five selected Amharic ambiguous words; these are አጠና*(eTena),* መሳል*(mesal),* መዛዝት*(me`sa`sat),* መጥራት*(metrat),* and ቀረጸ*(qereSe).* A total of 1045 English sense examples for the five ambiguous words were collected from British National Corpus (BNC). Solomon used five selected unsupervised algorithms such as Simple k means, EM and agglomerative single, average and complete link clustering algorithms. In this research total of four experiments has been conducted. The first experiment was to check the effect of stemming and stop word removal, the second one was to investigate the effect of different context sizes, the third was to see the effect of sense distribution and the last experiment was to compare the accuracy of selected algorithms. The researcher achieved accuracy within the range of 65.1 to 79.4 % for Simple k means, 67.9 to 76.9 for EM and 54.4 to 71.1 for Complete Link clustering algorithms for the five ambiguous words.

Getahun [**10**]have conducted a research with the main objective of the research was to design a WSD (word sense disambiguation) prototype model for Amharic words using semi-supervised learning method to extract training sets which minimizes the amount of the required human intervention and produce considerable improvement in learning accuracy. The researchers incorporated unlabelled data by combining some seeds instances directly into the data representation (features), so that unsupervised algorithms can be directly applied for clustering based on instances similarity. They used a python program which they made it in line with their preprocessing tasks. The experiment of semi-supervised methods using bootstrapping algorithms was conducted on the five Amharic WSD datasets following semi-supervised clustering assumption. These words were *atena* (አጠና), *derese* (ደረሰ), *tenesa* (ተነሳ), *bela* (በላ) and *ale* (አለ). Separate data sets using the five ambiguous words were prepared for the development of this Amharic WSD prototype. The experiment showed that the average performance results of Adaboost, Bagging and ADtree algorithms are 84.90%, 81.25% and 88.45%. And the researchers concluded that Semi-supervised learning using bootstrapping algorithm performs better in their study and it is more adaptive on WSD for Amharic. They also found that, a window size of 3-3 can be a standard window size for Amharic WSD systems development.

Gashaw [**11**] has used unsupervised corpus-based approaches to Amharic word sense disambiguation. For the research he has collected 4000 untagged sense examples from the web. The researcher used simple K-means, EM, single link, CURE and BIRCH from partitional, model-base, agglomerative, and hybrid clustering algorithms. He has experimented 10 selected ambiguous Amharic words; Ale (አለ), Bela(በላ), Akebere(አከበረ),Atena (አጠና),Derese (ደረሰ),Melese (መለሰ), Amelekete (አመለከተ),Qetere (ቀጠረ),Tenesa (ተነሳ) and Metrat (መጥራት) using WEKA 3.7.9 package for disambiguation and for document preprocessing purpose he used python 3.1. To evaluate the system the researcher used manually tagged text and selected "class to cluster" evaluation mode. For the 4000 training dataset the average accuracy obtained was 65.1%, 58.5%, 66.6% and 85.1% using K-means, single link, CURE and BRICH algorithms respectively.

Feyisa [**34**]conducted a research work on Word Sense Disambiguation for Afaan Oromo using unsupervised corpus based approach. He tested five clustering algorithms (simple k means, hierarchical agglomerative: Single, Average and complete link and Expectation Maximization algorithms) in the existing implementation of Weka 3.6.11 package. Due to lack of sense annotated text, he collected a total of 1500 Afaan Oromo sense examples for selected seven ambiguous words namely ***sanyii, karaa, horii, sirna, qoqhii, ulfina and ifa.*** To make the sense example sentences ready for experimentation, the researcher has performed activities like tokenization, stop word removal and stemming. He used contextual co-occurrence feature which indicate word occurrence within some number of words to the left or right of the ambiguous word. The researcher conducted two major experiments; the first was evaluation of the effect of stemmed data versus un-stemmed which showed that stemming is good to improve the accuracy of the algorithms and the second was evaluated with the window size dataset of one-one to eight-eight find that window size three-three is enough for EM and complete link algorithm while four-four is enough for simple K Means algorithm to identify the sense of the words. "Cluster via classification" evaluation mode was used to learn the selected algorithms in the preprocessed dataset. In general, the system performs with average accuracy of 81.9% for simple K Means, 79.8% for EM and 76.1% for complete linkage respectively.

Samrawit [**35**] carry out a thesis to investigate the effectiveness of using semantic similarity measure in WSD for query expansion. The study presented approaches to determine the sense of

words in queries by using Amharic lexical resource. For this purpose the researcher constructed a simple Word-Net composed of words with their synonym and gloss definitions. The WSD performed with two methods; gloss to gloss and synset to gloss by comparing information associated with its synonyms and gloss definition with reference to Amharic Word-Net. The combination of the two disambiguation methods formulates the modified query and used for expanding the original query. Finally, the query expansion module is integrated with Information Retrieval system to show the enhancement of Amharic IR system performance. The method using synset for query expansion register performance of 59% F-measure. The study has reported that this method registered 6% improvement from original query.

The researchers conducted a study on selected words from similar word classes using machine learning techniques. Solomon [**8**], Getahun [**10**] and Solomon [**9**] used 5 words class and Gashaw [**11**] selected 10 words from verb word class. All the researchers come up with a promising result and Getahun [**10**] resulted the highest accuracy employing semi-supervised machine learning algorithm which is 88.45%. From their study a WSD model is developed for those selected words only. Samrawit [**35**] on the other hand, showed the performance of Lesk algorithm for WSD using Word-Net. In this thesis, the aim is to investigate a more generic approach for developing Amharic WSD system that would work for all-word classes by combining knowledge from a tagged corpus and lexical resource.

Summary

WSD is a historical task in NLP that involves the automatic assignment of meaning to words based on their context. There are two variants of the generic WSD tasks: the lexical sample task and the all-words task. Many approaches have been proposed for the task but the main ones are Knowledge based and corpus based approaches. The knowledge based approach uses external knowledge sources to disambiguate words and the corpus based approach on the other hand employee different machine learning techniques to automatically assign meaning to words. Different researchers conducted research on Amharic WSD, in this thesis a more generic approach for Amharic is investigated.

CHAPTER THREE

Amharic Language

Amharic language started to be used in Ethiopia as early as the 14th century. Now, Amharic is the working language of the FDRE. Ethiopia includes many ethnic groups with nearly 89 languages and approximately 200 dialects [**36**]. The Ethiopian languages are divided in to four major language groups, such as Cushitic, Omotics, Nilo-Saharan and Semitic. Among these Amharic belongs to Semitic family of language. It is the second most spoken Semitic language in the world, next to Arabic. Amharic is estimated to be the mother tongue of more than 17 million people, with at least additional 5 million second language speakers. The language of Amharic is spoken in the Ethiopian government, court system, and on all official documents [**36**].

3.1. Amharic Writing system

Modern Amharic has inherited its system of writing from ancient Arabic by way of the Geez(Ethiopic) of the old kingdom of Axum, which latter is still the classical and ecclesiastical language of Ethiopia. The root, then, of Amharic orthography, like those of the language itself, are Semitic, the characters being designed to express the typical Semitic sounds used in the speech of ancestor [**37**]. The language is related to Arabic and Hebrew languages. Unlike the Arabic and Hebrew, the Amharic writing system is written from left to right. Amharic writing system has no upper and lower case latter variations and has no conventional cursive (*i.e*. written in a connected letters) form [**38**].

Amharic has its own writing system, a semi-syllabic system called FIDEL [**38**]. When we say Amharic writing system is semi-syllabic because of the fact that represents a consonant followed by a vowel. Since there are seven vowels in Amharic, it follows that there are seven different ways of writing a given consonant, depending on what vowel accompanies it [**36**]. Amharic language imposes 34 primary characters, each representing a consonant and each having 7 variations in form to indicate vowel which follows the consonant giving 238 distinct symbols. Beside them there are also forty "diphthong characters" that contain a special feature usually representing labialization e.g. ፗ, ቧ.

3.2. Amharic word classes

There are debates among scholars in classifying Amharic words. Previous scholars have classified Amharic words in to eight categories, which are Nouns, verbs, pronoun, adverbs, conjunctions, exclamation, and adjectives. However Baye [**12**]has classified in to five as noun, verbs, adjective, conjunction and adverbs.

3.2.1. Amharic Noun

Nouns [**12**] may denote gender, number, definiteness, case, and direct object status by affixes: prefixes and suffixes, predominately suffixes. Amharic nouns have two genders; masculine or feminine gender. The masculine gender can also serve as neuter for in animate objects. But occasionally in animate objects are treated as feminine. In particular there are no rigid rules as to when this should be for the same word way sometimes can be made masculine and sometimes feminine, according to fancy of the speaker. There is also a special "diminutive use" of feminine gender, by which nouns. Normally masculine, are treated as feminine in order to introduce the idea of littleness. Suffixes are added to denote a masculine or feminine noun gender. Most Nouns are made plural by addition of the suffix *–occ("ኦች")* [**37**]. Some words inherited from Geez still use the Geez plural forms. These are found especially in religious and literary language [**37**]. There are words that denot more than one meaning in this Amharic word class. For instance words like "ፈተና", "ደረጃ", "አካል" or "ሃይል" are nouns with multiple meanings.

3.2.2. Amharic verbs

Verbs are the most essential part of speech in Amharic language next to noun [**39**]. The original idea of the verbs is exhibited as a thing of time, found in a certain conditions and undergoing or producing various actions and changes. Verbs are derived from roots and affixes to inflect person, number, gender, aspect, mood, voice and polarity. Verbs must agree with their subjects, but verbs agreement with objects is optional. Verbs are placed at the end of the sentence. The root of Amharic verb consists in a number of "root letters" or "radicals" (most commonly three). To indicate person, tense, mood etc the forms of these radicals can change, prefixes and suffixes also can be attached but the radical themselves remains. The simplest part of a verb is its 3rd masculine singular simple perfect tense. The conjugation of an Amharic verb divides into three" moods": the indicative Mood, imperative mood (ordering) and the infinitive mode (verbal nouns) [**12**]. Words like "አለ", "አከበረ", or "ተነሳ" are classified under the category of Amharic verb

which are ambigous.

3.2.3. Amharic Adjective

Adjectives are descriptive words that come before nouns and show the character, size, color and shape of the noun. For example in the phrase "ንበዝ ተማሪ" the word "ንበዝ" shows the character of the noun "ተማሪ" which indicates braveness of the student. Word that can replace the word "ንበዝ" and can describe the noun can be classified as Adjective [**12**]. Words like "ሃሰተኛ", "ከባደ" or "ደማቅ"are some words that are classified under this class and denote multiple meanings.

3.2.4. Amharic Adverb

Amharic adverb is a word class that comes with a verb in a sentence and magnifies the verb. The word class has no many words like other class, Baye listed some main Adverbs like gmNa (ግምኛ), jlNa (ጅልኛ), mnNa (ምንኛ), kfuNa (ክፉኛ), Endegena (እንደገና) and gena (ገና). In the sentence "አስቱር ትላንት ምሳ በላች" the word "ትላንት" is an adveb that indicates the time that the action "በላች" was take place. And any word that can replace the word "ትላንት" can be classified as an Adverb [**12**].

3.2.5. Amharic conjunction

A conjunction is a word used to linkclauses within a sentence. Amharic conjunctiondonot undertake change of forms for any grammatic reasons like number, gender, level and etc ;or can not be a base for derivation of new words. Based on this criterias words like sle(ስለ), wede(ወደ), ke(ከ), le(ለ) and etc are classified under the word class conjunction [**12**]. Amharic conjunctions can be either separable or inseparable. There are different categories of conjunction in Amharic language: copulative conjunction like ደግሞ(again or also); Adversative conjunctions like ግን("however" or "but"); disjunctions like ወይም ("either" "or"); cuasal which include: እንደ(that),ዘንድ(inorder to); conditional containing words like እንደሆነ (if) and conclusive conjunctions like ስለዚህ(therefore) [**39**].

3.3. Ambiguity in Amharic language

Getahun [**13**] has described sources of ambiguity in Amharic language. Even though it is difficult to distinguish the sources of ambiguity, he has presented about six different sources of ambiguities which are: phonological ambiguity, Lexical ambiguity, structural ambiguity, referential Ambiguity, semantic ambiguity and orthographic Ambiguity.

- **Phonological ambiguity:** structure may be ambiguous because of differences in the placement of pause within them. For instance in the phrase "**ደግ ሰው ነበረ**" the word "ደግ ሰው" can have two meanings depending on the placement of the pause like these:

    1. [**ደግ**]+[**ሰው**]ነበረ meaning: " he was a kind man"
       Kind man was

    2. [**ደግሰው**] ነበረ meaning: "They had made a preparation for a banquet.'
       banquet was

- **Lexical Ambiguity:** there are different sources which are considered as a factor for lexical elements that could cause ambiguity in a given construction**.** Getahun discussed three of them as an example:

    - **Categorical Ambiguity:** Lexical elements can be considered as categorically ambiguous if they have identical phonological form but different word class. For example in the sentence "አከርማ ሰጠችኝ" the word አከርማ is ambiguous between the interpretations:

        a. She gave me **<u>Akirma</u>** (a kind of grass).

        b. She gave me something **<u>after delaying it for some time</u>**.

    - **Homonymy**: There are a number of lexical items with the same phonological form but different in meanings. Such forms lead to ambiguity in structures like the sentence አርብ ጠፋ is ambiguous because the word "አርብ" can mean "Friday" or " weavers of frame"

    - **Homophonous affixes**: homophonous affixes are another source of ambiguity in the language which is considered as factor of lexical elements based ambiguity. Consider the sentence "ቤቱ ፈራረሰ" which has the following two readings:

      ቤት+ኡ ፈራረሰ

        a. The house is destroyed.

        b. His house is destroyed.

      The sentence is ambiguous because the suffix /-ኡ/ serves as a definite article or as a third person masculine marker.

- **Structural Ambiguity**: the most common type of ambiguity in Amharic language is structural ambiguity which is related with the way syntactic constituents are organized. This ambiguity arises when a constituent of a structure has more than one possible position. Structurally ambiguous sentences have one surface realization that derives from more than one underlying representations. The sentence "የአቢሲኒያ ታሪክ አስተማሪ" has two readings:

  a. A person who teaches Abyssinian history.

  b. An Abyssinian who teaches history.

  The ambiguity of this sentence can be explained in terms of differences in the structural organization of the sub-constituent /ታሪክ/:

  a. [[[የአቢሲኒያታሪክ][አስተማሪ]]] ⟶ 'Abisynian history teacher'

  NP NP N

  b. [[የአቢሲኒያ][ታሪክ አስተማሪ]]]A teacher who teaches Abyssinian history.

  NP N NP N

- **Referential ambiguity**: Ambiguity may arise when a pronouns has more than one possible antecedents, thus having as many readings as there are antecedents. Let us consider the following example sentence:"ካሳ ስለተመረቀ ተደሰተ" has the following three possible readings:

  a. $kassa_i$ was pleased because he_i graduated.

  b. $kassa_j$ was pleased because he_i graduated

  c. He was pleased because kassa graduated

- **Orthographic Ambiguity**: some structures could be ambiguous because of orthographic reasons since the system does not show distinctions between geminate and non-geminate

sounds. The intended meaning of a word could be understood from the context. The examples like "መኪናው ይሰራል" is ambiguous between the following readings:

- the car will be repaired.
- The car works

- **Semantic ambiguity:** are ambiguities caused by polysemic, idiomatic and metaphorical constituents.
    - Example for ambiguity caused due to polysemic constituents is "መብራቱ ጠፋ". This statement is ambiguous between the interpretation of :

      a. The light went off

      b. Mebratu(a person) disappeared
- Idiomatic constituents:

  በሬ ወለደ can be interpreted as:

  a. an ox gave birth to a calf

  b. an idiomatic expression' unheard of' or 'impossible to happen'.
- Metaphors may have literal or non-literal (metaphoric) senses as in the following phrase "**አራስ ነብር**", which can be interpreted as:
    a. hot tempered
    b. leopard with new born cubs

Summary

Amharic language is working language of FDRE. The Language is from semetic family that has semi-syllabic writing system. Amharic has five word classes nouns, verbs, adjectives, adverbs and conjunctions. And there are six types of ambiguities in the language which are Phonological ambiguity, Lexical ambiguity, Structural ambiguity, Referential ambiguity, Orthographic ambiguity and Semantic ambiguity.

CHAPTER FOUR

Word Sense Disambiguation System Design and Architecture

Ever since the WSD field inception, different approaches have been proposed by different researchers [**1**]. Among the different WSD approaches one of the variations of Lesk algorithm which is corpus based Lesk algorithm is implemented. The corpus based Lesk algorithm uses manually annotated test examples and weight from Word-Net. For this research Amharic Word-Net is used as knowledge source. There is no well-constructed Amharic Word-Net; the Word-Net constructed by [**35**] is adopted with some modification in order to make it suitable for our experiment. In addition to the Word-Net manually tagged corpus also used in order to train the system.

The system accepts input and disambiguates a given ambiguous word in the text by measuring similarity between the senses of the ambiguous word and words around it. The similarity indicates the closeness or separation of a given objects. There are different ways to measure the degree of similarity; however the most common ones: cosine similarity measure and Jaccard coefficient [**19**] are selected for this study. In the next sub sections, the Architecture of the WSD system, Word-Net preparation, corpus preparation, the two similarity measures and the WSD system evaluation method are discussed in detail.

4.1.The Architecture

As shown in the figure 3.1 below, the system first accepts text containing ambiguous words as an input and then preprocesses the text. The preprocessing task involves tokenization, stop word removal, stemming and normalization. After that the system use similarity measure to disambiguate the words in the input text. To do so the system uses information from tagged corpus and Amharic Word-Net.

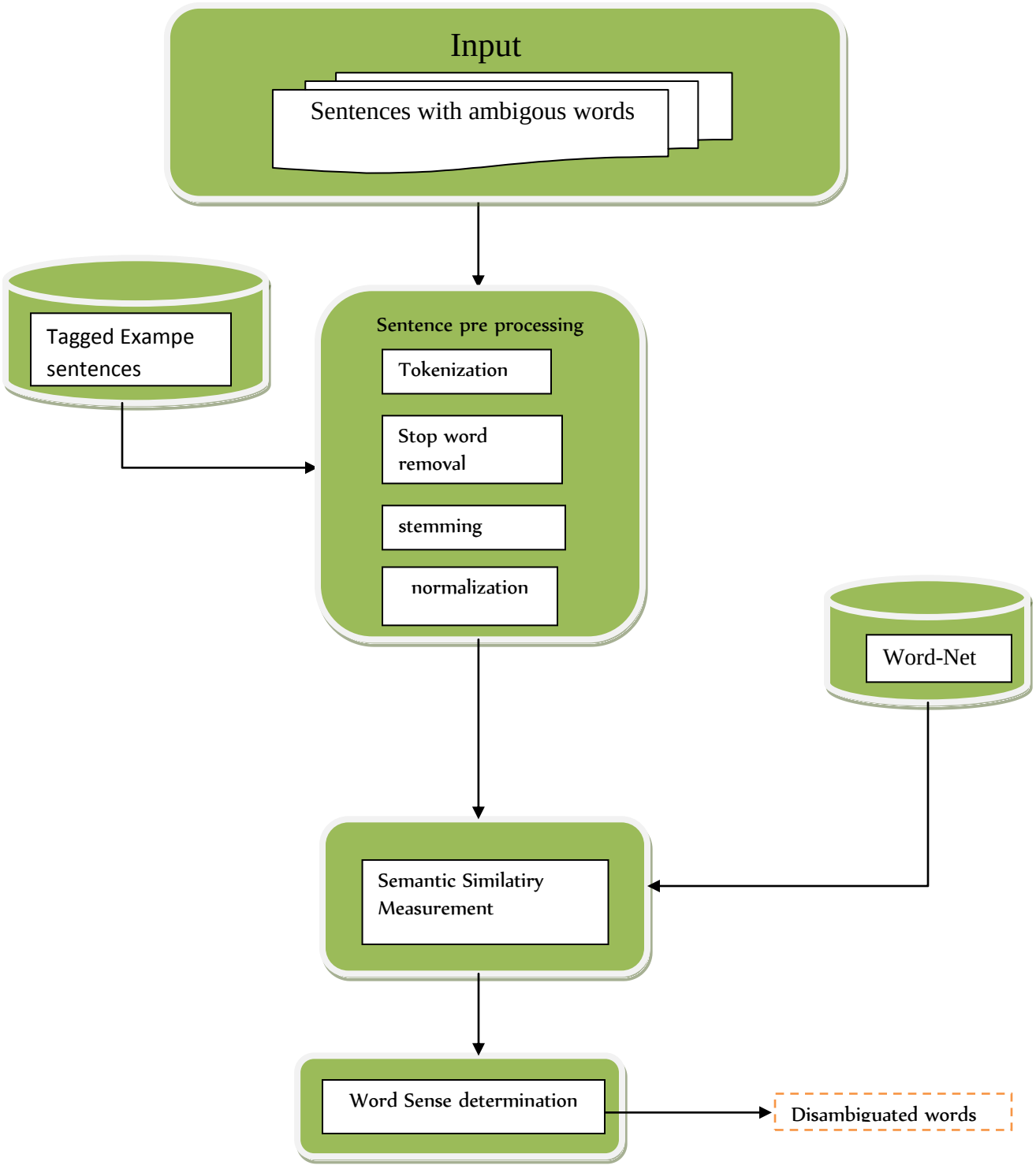


Figure 4.1: WSD system Architecture

4.2. Word-Net preparation

The origin of Word-Net is to build a lexical-conceptual model, consisting of both lexical units and the relations between such units, structured into a relational semantic network [**40**]. For Amharic language we don't have well-constructed Word-Net, so we adopted the Word-Net from Samrawit [**35**], the Word-Net consists of about 39 words with their synsets and gloss definitions for each sense of a given word.

By using Samrawit's Word-Net we have constructed a simple Word-Net by adding some lexical units and their relations in order to make it suitable for our experiment. We have used Amharic Dictionary that was prepared by Ethiopian Linguistics and research Institute [**16**], and Zergaw's Amharic Context Dictionary [**41**]. There are different relationships between words, for this research we have only used synonymy relationship to identify the meaning of an ambiguous word.

The structure of the Word-Net is the same as Samrawit's Word-Net. The Word-Net is composed of three major elements which are the words, their synsets and gloss definition, see figure 4.2. The Word-Net entries are organized in such a way that, after each word there is "@" sign that separates it from its senses and each senses are separated by semi-colon. The colon is used to separate the synonyms and the gloss definations. The Word-Net is organized this way in order to make it suitable for coding.

ፈተና@ፍተሻጥያቅ:አንድሰውስለአንድየተወሰነነገርማወቅአለማወቁንለመገንዘብትምህርትቤትክፍልውስጥበፅሁፍወይምበቃ ልየሚቀርብጥያቄ;ችግርመከራስቃይ:በልዮልዮነገርመቸገርመሰቃየት

ክፍል@*መመሪያ* ክፍል:ትምህርት ቤት የሚገኙት እያንዳንድ የመማሪያ ክፍሎች;አንድ ከመሆን የተከፈለ:አንድ ከመሆን ብዛት ያገኘ የአካል የሰውነት የሀገር ክፍል;ትምህርት ደረጃ:የተወሰነ ትምህርት በተወሰነ ደረጃና ይዘት ተመጥኖ የሚሰጥበት የትምህርት ምድብ; የስራ አካል:በአንድ መስሪያ ቤት ውስጥ የሚገኝ የስራ አካል

ዘር@ወገን ዘመድ ትውልድ ተወላጅ: የትውልድ ሀረግ የሚታወቅበት በደም ግንኙነት የተሳሰረ የዝምድና ሰንሰለት; ከፍሬ የሚገኝ: አስፈላጊ ነገሮች ሲሞሉለ(አፈር ውሃ አየር *ሙቀት*) የሚበቅል የእፅ መራቦ

ሃይል@ብርታት ጥንካሬ ጉልበት:አንድን ተግባር ለማከናወን የሚያስችል ጉልበት አቅም;ቡድን:አንድን ተግባር የሚፈፅም ቡድን;አስገዳጅ:ዘዴ ያልተከተለ አስገዳጅ ግፊት

Figure 4.2: Sample Amharic Word-Net

The set of near-synonyms for a Word-Net sense is called a **synset** (for **synonym set**); synsets are an important primitive in Word-Net. As it can be seen from the figure 4.2, the synsets of the words are placed before the colons in the Word-Net. For example the entry for the word "ፈተና" includes synsets like "ሙከራ ፍተሻ ጥያቄ ምርመራ" and "ችግር መከራ ስቃይ"; and for the word "ሃይል" the synsets are "ብርታት ጥንካሬ ጉልበት","ቡድን" and "አስገዳጅ ".

Other part of the Word-Net is the gloss definition which is located after the colon. The gloss is a dictionary styled textual definition of the synset possibly with a set of usage examples that express the concepts in each synsets. The gloss definition for the first sense of "ፈተና"which is "ሙከራ ፍተሻ ጥያቄ ምርመራ" is "አንድ ሰው ስለአንድ የተወሰነ ነገር ማወቅ አለማወቁን ለመገንዘብ ትምህርት ቤት ክፍሎች ውስጥ በፅሁፍ ወይም በቃል የሚቀርብ ጥያቄ" and for the second one "ችግር መከራ ስቃይ" is "በልዮ ልዮ ነገር መቸገር መሰቃየት".

4.3. Corpus Preparation

Since corpus based Lesk algorithm relies on manually annotated example sentences in addition to Word-Net or other lexical resources, we need example sentences for the different senses of the selected words. Therefore, tagged corpus that contains example sentences has been prepared for senses of 17 selected Amharic words from 3 Amharic word classes in order to train and test the WSD system prototype. A total of 9 nouns, 3 verbs, 3 adjectives and 2 adverbs are selected randomly from Amharic Dictionary [**16**] as presented in table 3.1.

Eventhough most of the selected words have more than 2 senses, we have selected atmost 3 senses for each words. We couldn't use all the possible senses of the words because, some senses of the words appear rarely. So we selected senses that appear dominantly. Example sentences are composed for the selected senses. The example sentences are collected from news, reports, spritual manuscripts, posts from governmental and non-governmental organizations as well as from different poeple opinions.

The corpus is preprocessed before being used by the system. The preprocessing involves tokenization, stop word removal, stemming and normalization. A tool for tokenization, stop word removal and normlization is developed in this thesis using python code. For the stop word removal process, the stop words are taken from the list identified by Amanuel [**42**], who

conducted a research on probablistic information retrieval system for Amharic language. In addition a code for stemming process is also adopted from Amanuel [**42**] with some modification.

The corpus contains 100 example sentences for each sense of the selected words. We used only 100 example sentence because we were able to collect up to 100 sentences for each senses. The corpus is organized in to different documents in such a way that a document have set of example sentences for a single sense of a word. For instance for the word **መንገድ** we have two documents:

- document1- containing example sentences of sense1: ብልሃት ዘዴ አሰራር
- document2- containing example sentences of sense2: መሄጃ መመላለሻ መተላለፊያ

Table 4.1 shows the number of possible senses of the selected word which is taken from Amharic dictionary [**16**], senses selected for the experiment and number of sentences in the corpus for the selected.

Word classes	Word	No. of possible senses	Senses	No. of example sentences
Nouns	አካል	3	የሰውነት ክፍል	100
			ክፍል አባል	100
			ማህበር	100
	እድገት	2	የአካል ክፍል መጨመር	100
			የተሻለ ማግኘት	100
	ዘር	5	ትውልድ ዝርያ	100
			ከፍሬ የሚገኝ	100
	ድካም	2	ዝለት	100
			ጥረት	100
	መንገድ	3	ብልሃት ዘዴ አሰራር	100
			መሄጃ መመላለሻ መተላለፊያ	100
	ፈተና	2	ሙከራ ፍተሻ ጥያቄ ምርመራ	100
			ችግር መከራ ስቃይ	100
	ልጅ	6	የአብራክ ክፉይ	100
			ህፃን	100
	ድምፅ	2	ጩሀት	100
			ውሳኔ ማሳወቅ ምርጫ	100
	ዋጋ	3	ፋይዳ ጥቅም	100
			ተመን	100
Verbs	ደረሰ	8	ተፈጸመ ሆነ	100
			በቃ	100
	ቆረጠ	12	ገመደ	100
			ወሰነ	100
	አከበረ	4	በአለ አከበረ	100
			ከፍ አደረገ	100
Adjectives	ሃሰተኛ	2	ውሸት የሚናገር	100
			ትክክለኛ ያልሆነ	100
	ደማቅ	2	የነላ	100
			የሞቀ	100
	ትኩስ	2	ሙቅ	100
			አዲስ ለጋ	100
Adverb	በቀስታ	2	በዝግታ በርጋታ	100
			በዝቅተኛ ድምጽ	100
	ወደፊት	3	ለሚቀጥለው ጊዜ	100
			ከፊት አቅጣጫ	100
			Total number of sentenses	3500

Table 4.1: Example sentences in the corpus

4.4. The Amharic WSD system

Knowledge based methods for WSD are usually applicable to all words in unrestricted text. The objective of knowledge-based or dictionary-based WSD is to exploit knowledge resources (such as dictionaries, thesauri, ontologies, collocations, etc.) to infer the senses of words in context. Among the knowledge based technique's overlap of sense definitions,Selectional restrictions, and structural approaches, in this study overlap of sense definitions is used, which is a simple and intuitive knowledge-based approach that relies on the calculation of the word overlap between the sense definitions of two or more target words. This approach is named Lesk algorithm [**3**]. As mentioned earlier corpus-based Lesk algorithm is employeed for this study.

4.4.1. Corpus-Based Lesk Algorithm

As described in the literature review section 2.6 , the corpus-based Lesk algorithm is used to disambiguate one target word provided that a set of annotated training examples is available [**18**]. The algorithm uses weight from a tagged corpus and lesk algorithm. To compute the weight of senses from the tagged corpus, similarity measures are used. For this purpose, set of training example sentences for each senses of a word are considered as a single document and each document j may be represented as [**40**]:

$$D_j=\{tf_{1,j}, tf_{2,j}, tf_{3,j},.....tf_{i,j}\}$$

Where *tfi, j* is the *term frequency* of the word $w_i$in document *j*. Similarly, an input text may be represented as:

$$T=\{ tf_{1,t}, tf_{2,t}, tf_{3,t},.....tf_{i,t}\}$$

Because we deal with annotated corpus, we need to add other factor which is inverse document frequency. Inverse document frequency *idf* assigns more weight to words which are more specific for the given document. IDF can be calculated as: [**19**]

$$idf_j = \log \frac{N}{df_j} \quad \text{.....(4.1)}$$

Where *N* is the total number of documents in the collection and df_j is the number of documents containing the word *wj*. So the weight of the sense can be computed by using the tf-idf of words contained in the document containing example sentences of that sense as an input and measuring the similarity between the document and the input text [**40**].

4.4.2. Similarity measures for WSD

Similarity measure reflects the degree of closeness or separation of a document and the given statement. Variety of different similarity measures can be used to calculate the similarity between the input text and the documents containing tagged example sentence. This similarity measure are used for WSD. A characteristic of a similarity formula is that the results of the formula increase as the items become more similar. The value is zero if the items are totally dissimilar. The similarity measures employed to extract information from the tagged corpus. Among the different types of similarity measures, cosine similarity and Jaccard coefficient similarity measures are chosen and implemented because they are most popular similarity measures and can be easily implemented for WSD [**19**].

Cosine Similarity

The simplest approaches are based on the bag-of-words (multiset of words) model. In such a model the correct sense of the input text is determined based on the cosine of appropriate vectors. Cosin similarity is a widely implemented metric in information retrieval and related studies. This metric models a text document as a vector of terms. By this model, the similarity between the words can be derived by calculating cosine value between documents and the input text term vectors [**43**].

The cosine measure calculates the similarity between two documents as the cosine of the angle between their corresponding word vectors [**43**]. An important property of the cosine similarity is its independence of document length. As the Cosine approaches "1," the two vectors become coincident. If the two are totally unrelated, then they will be orthogonal and the value of the Cosine is "0." What is not taken into account is the length of the vectors [**19**].

The Cosine similarity between input text and a document is calculated as:

$$sim(\vec{T}, \overrightarrow{d_J}) = \frac{\sum_{w \in T,d} tf_{w,j}(idf_w)^2}{\sqrt{\sum_{i=1}^{n}(tf_{i,T}idf_i)^2} \times \sqrt{\sum_{i=1}^{m}(tf_{i,j}idf_i)^2}} \quad \text{.....(4.2)}$$

Where,

- $tf_{w,T}$ is term frequency of the word w in Text T but $tf_{i,T}$ is term frequency of word w_i in the input text.

- $tf_{w,j}$ term frequency of the word w in document d_j.
- idf_w is inverse document frequency of the word w in document j.
- *n* and *m* are number of the different words in Text and document j respectively.

The numerator represents the dot product (also known as the inner product) of the vectors (*d*1) and (*T*), while the denominator is the product of their Euclidean lengths [**44**].The above equation can be used to calculate the best word sense as following: suppose the word we wanted to disambiguate in an input text contains two senses, which means we have two documents to compare with the sentence. The cosine similarity between the query and each of the two documents will be computed and the sense represented by the document that resulted highest value will be returned as the best sense of the ambiguous word.

Jaccard co-efficient

The Jaccard co-efficient, which is sometimes referred to as the Tanimoto coefficient, is a statistic used for comparing the similarity of two sample sets and is defined as size of intersection divided by size of union on sample data sets [**45**]. For text document, the Jaccard coefficient compares the sum weight of shared terms to the sum weight of terms that are present in either of the two documents but are not the shared terms [**44**]. The Jaccard coefficient between documents i and input query j can be calculated as:

$$SIM(DOC_i, TEXT_j) = \frac{\sum_{k=1}^{n}(DOC_{i,k} \times TEXT_{j,k})}{\sum_{k=1}^{n} DOC_{i,k} + \sum_{k=1}^{n} TEXT_{j,k} - (DOC_{i,k} \times TEXT_{j,k})} \quad \text{.....(4.3)}$$

where,

- $DOC_{i,k}$ is the tf-idf of word in document i
- $TEXT_j$ is the tf-idf of word in input text j
- n is the total number of descrite terms in either the document or the query

The Jaccard coefficient is a similarity measure that ranges between 0 and 1. It is 1 when the DOC_i=text and 0 when DOC_i and term are disjoint, where 1 means the two objects are the same and 0 means they are completely different. The Jaccard coefficient will be used in the same way as that of Cosine similarity to identify the best sense of an ambiguous word in a give sentence.

4.4.3. Lesk Algorithm

The Lesk algorithm is one of the first algorithms developed for semantic similarity [**18**]. The algorithm uses dictionary entries for each possible word senses. The Lesk algorithm, in which the most likely meanings for the words in a given context are identified based on a measure of contextual overlap among dictionary definitions pertaining to the various senses of the ambiguous words [**1**]. Given a two word context (*w*1, *w*2), the senses of the target words whose definitions have the highest overlap (i.e., words in common) are assumed to be the correct ones. Formally, given two words *w*1 and *w*2, the following score is computed for each pair of word senses *S*1 ∈*Senses*(*w*1) and *S*2 ∈*Senses*(*w*2):

$$scoreLesk(S1, S2) =| gloss(S1) \cap gloss(S2) |,$$

For example, for the text "*መማሪያ* ክፍል" both having three senses:

"*መማሪያ*" has three senses

- *መጽሃፍ ደብተር:ትምህርት ለመማር የሚያስፈልጉ መሳሪያዎች*;
- *መማሪያ ትምህርት ቤት: ተማሪዎች የሚማሩበት የትምህርት ቦታ ክፍል*;
- *ርኅራሄ ሃዘኔታ ይቅርማለት: ያጠፋ የበደለ ሰዉ ምህረት ይቅርታ እንዲደረግለት*;

"ክፍል"

- *መማሪያ ክፍል ክላስ:ትምህርት ቤት የሚገኙት እያንዳንድ የመማሪያ ክፍሎች*;
- *የተከፈለ:አንድ ከመሆን ብዛት ያገኘ የአካል የሰውነት የሀገር ክፍል*;
- *መለያ ምራአፍ:መጻህፍት በፊልሞች ላይ ያሉ መረጃዎች በአንቀጽ ወይም በምእራፍ ሲከፈል*;

Lesk(*መማሪያ*,ክፍል)	sense1:*መማሪያ*	sense2:*መማሪያ*	sense3:*መማሪያ*
sense1:ክፍል	1	3	0
sense2:ክፍል	0	1	0
sense3:ክፍል	0	0	0

Table 4.2: Example for Lesk algorithm

To determine the sense of "ክፍል" the algorithm counts the number of words that the gloss definitions have in common and the sense with the highest overlap is considered as the correct sense for the ambiguous word. As shown in table 4.1, there are 9 possible combinations of senses and among those combinations the first definition for "ክፍል" and the second definition for "*መማሪያ*" have the largest overlap among all possible sense combinations, with three words in common:ትምህርት, *መማሪያ*, and ክፍል, and therefore the meaningselected by the Lesk algorithm for the given pair "*መማሪያ* ክፍል" will be "*መማሪያ*ክፍልክላስ".

Since corpus based Lesk algorithm involves combination of information from tagged example with Word-Net, the Lesk algorithm is combined with the two similarity measures. For this the result from the Lesk algorithm is added with the score from the similarity measures. The result from the similarity measures ranges between 0 up to 1, while the Lesk algorithm may come up with score greater than 1. So while combining the two, the result from the similarity measures become insignificant, therefore there is a need to normalize the score from Lesk algortihm. In order to normalize logarithm of the result is used. In cases when Lesk algorithm resulted 0, the result will be replaced by negative one (-1). Consequetly for combining, logarithm of the result from the Lesk algortihm is added with the results of the similarity measures. Equation 4.4 presents the formula for combining the Lesk with similarity measures.

$$C(s) = \log L(s) + SIM(s) \quad \text{.....(4.4)}$$

Where C is combination of the two results for sens s, L the result from the Lesk algorithm of sense s, S is the result from the similarity measures of sense s.

4.5.System Evaluation

The goal of any WSD system is to identify sense of ambiguous words in a given context. The system is evaluated based on number of times that the system gives the correct sense of a particular ambiguous term given different sentences. The accuracy of the system is calculated as follows:

$$Accuracy = \frac{n}{N} \times 100\% \quad \text{..... (4.5)}$$

Where, n is the number of sentences correctly annotated, and N is the total number of sentences in the test examples

Summary

In this chapter the design and architecture of the WSD system is discussed. And, the methods and techniques used to develop the WSD system prototype are presented. The main parts of the WSD system are Word-Net, tagged corpus and the similarity measure. The chapter discussed the Word-Net and tagged corpus preparations and the two similarity measures: Jaccard coefficient and the Cosine similarity that are employed to automatically identify the sense of a word in detail. And in the next chapter, the result, experimental findings and challenges of the study are presented.

CHAPTER FIVE

Experimentation and Discussion

The main goal of any WSD research is to resolve ambiguity of different senses listed in a dictionary, thesaurus or another source of a given word [**46**]. Accordingly, this research attempts to find a way to automatically determine the sense of ambiguous Amharic words in a given context. For this purpose, an Amharic WSD system prototype is developed. As described in the literature review section there are different approaches for WSD, however the corpus based lesk algorithm to have been selected to develop the system.

In this thesis, the WSD system prototype developed has three main parts: the Word-Net, tagged corpus and the semantic similarity measure. As discussed in the previous chapter, the sense of an ambiguous word in a given sentence can be identified by measuring the similarity between the example sentences of the word and the input sentence. To calculate their similarity cosine similarity measure and Jaccard coefficient are used.

In this study total of 5 experiments have been conducted on selected 17 ambiguous words.The first experiment is done to test the performance of the cosine similarity measure for WSD, the second one is on Jaccard coefficient similarity, the third experiment is conducted to test the Lesk algorithm and the last two experiments done on corpus based algorithm using cosine similarity and Jaccard coefficient similarity measure. The result and detail discussion of the experiments are presented in the preceding sections.

5.1. Preparation of Test Dataset

Five different expiriments have been carried out to test the performance of the developed system. For testing task test dataset is prepared for the selected words. The test dataset is composed of 10 sentences for each sense of the 17 ambiguous Amharic words. Table 5.1 shows test sentences for the two senses of the word "እድገት:"

Word : እድገት	**Test sentences**
የአካል ክፍል መጨመር	ልጆች ጤና አካል እድገት ወሳኝ ነው
	የአእምሮ እድገት ውሱንነት አለበት
	የሰውነት እድገት ለውጥ ማሳየት ጀመረ
	አንጎል እድገት ላይ ተፅኖ አለው
	የህጻኑን የእድገት ሒደትን በዝርዝር አስቀመጠ
	የግንዛቤና የማህበራዊ ክህሎት እድገት አሳየ
	የሳንባ እድገት 4 ደረጃ አለው
	ቫይታሚንዲለአጥንትእድገትያስፈልጋል
	በማሕፀን ውስጥ ጽንስ እድገት ያደርጋል
	እድገት ዝግመት ያለበት ተማሪ እንክብካቤ ያስፈልገዋል
የተሻለ ማግኘት	ለስራ እድገት ውድድር ተደረገ
	የሕክምናው መስክ እድገት እያሳየ ነው
	ዝውውር ና ደረጃ እድገት አገኘ
	ለኢኮኖሚ እድገት ምቹ ሁኔታ ይፈጥራል
	ዜጎች እድገት ና ብልፅግናን ማስተጓጎሉ ተገለፀ
	ጠንካራ እድገት ና ለልማት ያስፈልጋል
	ቱሪዝም እድገት አስተዋፅኦ አለው
	የትምህርት ዘርፉ እድገት ማስመዝገብ ጀመረ
	ከፈጣን የቴክኖሎጂ እድገት ጋር አብሮ መራመድ አለበት
	መንገድ ለአንድ ሀገር እድገት ወሳኝ ነው

Table 5.1: Sample test sentences for the word "እድገት"

5.2. Experimental procedure

In this study, total of five experiments are done to test the performance of the WSD system. As mentioned earlier in the previous chapter, we have selected corpus based Lesk algorithm for developing Amharic WSD. This algorithm identify senses by combining the weight from Lesk algorithm and the sample examples. However since we don't have well-structured Word-Net, the performance of similarity measures without the Lesk algorithm is tested to see the possibility of using similarity measures alone for WSD. For this purpose, the performance of 4 similarity measures: Euclidean distance, Cosine similarity, Jaccard coefficient and Dice were tested.

The WSD based on Euclidean distance resulted a very small accuracy which is 12% and the WSD based on Dice come up with a result the same as that of Jaccard coefficient. As a result, the experiments in this study focused on the Cosine similarity measure and Jaccard coefficient. So the first two experiments are done to test the performance of the two selected similarity measures: individually. Then Lesk algorithm is tested and in the last two experiments the performance of combination of the similarity measures with Lesk algorithm is checked.

5.2.1. Experiment 1: Applying cosine similarity for WSD

As discussed in the previous chapter section 4.5, Cosine similarity is a measure of the cosine angle between two vectors. The two vectors in our case are: the documents containing example sentences; and the input sentence containing the ambiguous word. The documents and the input texts are represented by the weight of the words incorporated in them which is the tf-idf of the words. The cosine angle between the given statement and the documents containing the sense examples of the ambiguous words are measured and the one with highest score is taken as the sense of the word. For example, to disambiguate the word "እድገት" in an input text "**የአእምሮ እድገት ውሱንነት አለበት**":

$SIM(T_{\text{የአእምሮ እድገት ውሱንነት አለበት}}, d_{\text{የአካል ክፍል መጨመር}}) = 0.7286$

$SIM(T_{\text{የአእምሮ እድገት ውሱንነት አለበት}}, d_{\text{የተሻለ ማግኘት}}) = 0.526$

As we can see the input statement is more similar to the document containing the example sentences of the first sense of the word "እድገት". So the algorithm decides that the sense of the word in the give context is "**የአካል ክፍል መጨመር**" as shown in figure 5.1.

```
ኣረፍተ ነገር ያስገቡ: የአእምሮ እድገት ውሱንነት አለበት
ትርጉሙን የፈለጉትን ቃል ያስገቡ: እድገት
የስገቡት ቃል ትርጉም :   የአካል ክፍል መጨመር
>>>
```

Figure 5.1: Example of WSD using Cosine similarity measure

As we can see from the result summary table 5.1, the accuracy of applying cosine similarity for WSD ranges between 55%-100% and the average accuracy of the similarity is 81.94% for nouns , 88.67% for verbs, 90% for adjectives and for adverbs 77.5%. The total average accuracy of the

WSD system based on cosine similarity is 84.52%. WSD based on this similarity measure produced highest accuracy for adjectives and lowest accuracy for adverbs. And the similarity resulted the highest performance for the words "ዘር" and "ቆረጠ", which is 100% and come up with the least accuracy for the word "ልጅ", that is 55%.

The WSD based on Cosine similarity measure resulted high for the words that have senses that donot appear frequently with the same words. And produce least accuracy for the word like "ልጅ" which have senses that share common collocated words. The same is true for the least performance of this approach for adverbs.

Accuracy of cosine similarity measure			
Word classes	**Words**	**Accuracy**	**Average accuracy**
Nouns	ፈተና	85.5%	81.94%
	እድገት	80%	
	መንገድ	95%	
	ዘር	100%	
	አካል	75%	
	ዋጋ	80%	
	ልጅ	55%	
	ድካም	73%	
	ድምፅ	90%	
Verbs	ደረሰ	88%	88.67%
	ቆረጠ	100%	
	አከበረ	78%	
Adjectives	ሃሰተኛ	92%	90%
	ደማቅ	88%	
	ትኩስ	90%	
Adverbs	በቀስታ	65%	77.5%
	ወደፊት	90%	
		Total average accuracy	84.52%

Table 5.2: Accuracy of Cosine similarity measure for WSD

5.2.2. Experiment 2: Applying Jaccard coefficient for WSD

To compute the similarity between the input text and the documents the Jaccard coefficient have been employed which measures similarity as the intersection divided by the union of the objects. The Jaccard coefficient similarity ranges between 0 and 1. The document with the highest Jaccard coefficient value is considered more similar to the query than the other. For instance the sense of the ambiguous word "መንገድ" in the query "ምርት በባህላዊ መንገድ አመረቱ" can be identified by using Jaccard coefficient similarity measure as follows:

$SIM(T_{\text{የከፍተኛ ትምህርት መግቢያ ፈተና ወሰዱ}}, d_{\text{ፍተሻ ጥያቄ}}) = 0.012113$

$SIM(T_{\text{የከፍተኛ ትምህርት መግቢያ ፈተና ወሰዱ}}, d_{\text{ችግር መከራ ስቃይ}}) = 0.00127824$

The result from the similarity measure indicates that the document containing example sentences of "ብልሃት ዘድ አሰራራ" has highest jaccard coefficient value, so the algorithm decides that the sense of "መንገድ" in the given context is "ብልሃት ዘድ አሰራራ".

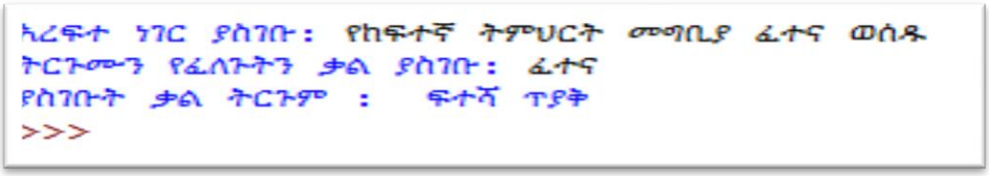


Figure 5.2: Example of WSD using Jaccard Coefficient

Summary of the performance of the Jaccard coefficient is presented in the table5.2. The accuracy of the similarity measure ranges from 55% - 100% with average accracy of 83.68%, 90.67%, 87% and 82.5% for nouns, verbs, adjectives and adverbs respectively. And the total average accuracy of Jaccard coefficient is 85.96%. The Jaccard coefficient similarity generated least performance for adverbs as that of cosine similarity and performs high for the verb word class. And the similarity measure produced the highest performance for the words "ዘር" and "ቆረጠ" and the least performance for the word "ልጅ" as that of the cosine similarity.

Like that of the WSD based on Cosine, the Jaccard is also affected by words that have senses that share common collocated words. In most of the cases Jaccard coefficient performs the same as that of Cosine similarity, however the WSD based on this similarity produced 1.44% better accuracy than the Cosine similarity. This is because of the fact that both similarity measures have normalization factor, the denominator in the Cosine formula is invariant to the number of terms in common and produces very small numbers when the vectors are large and the number of common elements is small. Jaccard change the normalizing factor in the denominator to account for different characteristics of the data [**43**]. This may be the reason for the better performance of Jaccard coefficient than Cosine similarity.

Accuracy of Jaccard coefficient			
Word classes	**Words**	**Accuracy**	**Average accuracy**
Nouns	ፈተና	86%	83.68%
	እድገት	90%	
	መንገድ	90%	
	ዘር	95%	
	አካል	80%	
	ዋጋ	85%	
	ልጅ	55%	
	ድካም	84%	
	ድምፅ	90%	
Verbs	ደረሰ	88%	90.67%
	ቆረጠ	100%	
	አከበረ	84%	
Adjectives	ሃሰተኛ	92%	87%
	ደማቅ	88%	
	ትኩስ	81%	
Advers	በቀስታ	75%	82.5%
	ወደፊት	90%	
		Total average accuracy	85.96%

Table 5.3: Summary of accuracy of Jaccard coefficient for WSD

5.2.3. Experiment 3: Applying Lesk algorithm for WSD

The third experiment is conducted to test the performance of applying Lesk algorithm for WSD. Here for employing Lesk algorithm, the system searches for the lexical entries of the collocated words in the Word-Net. If there is a lexical entry for the word, the overlap between the gloss definitions of each senses of the target word and the words that are collocated with the target word is counted. The sense of the target word with the highest overlap is considered as the correct sense of the word in the given context. For instance to disambiguate the word "ፈተና" in a sentence "የከፍተኛ ትምህርት መግቢያ ፈተና ወሰዱ", the lesk algorithm first looks for words that have lexical entries in the Word-Net. We have Word-Net entries for the words "ትምህርት", "መግቢያ" and "ፈተና". Here, the Word-Net entries for the words are presented to illustrate this method.

The Word-Net entry for the word:

- "ትምህርት" is: "እውቀት:ትምህርት ቤት በመግባት ና በመማር ወይም ከስራ ልምድ የሚገኝ እውቀት;
- "መግቢያ":

•sense1 -በር:ወደ ውስጥ የሚያስተላልፍ በር;

•sense2- መጀመሪያ: ወደ ሌላ ክፍል ትምህርት ደረጃ መዘዋወሪያ ፈተና

The word "ፈተና" has two senses:

- sense1- ፍተሻ ጥያቅ:አንድ ሰው ስለአንድ የተወሰነ ነገር ማወቅ አለማወቁን ለመገንዘብ ትምህርት ቤት ክፍል በፅሁፍ ወይም በቃል የሚቀርብ ጥያቄ
- sense2- ችግር መከራ ስቃይ:በልዮ ልዮ ነገር መቸገር መሰቃየት

The word "ትምህርት" has "ትምህርት", "ቤት", and "እውቀት" in common with sense 1 of "ፈተና" and has no common word with sense 2 of "ፈተና". And since the word "መግቢያ" has two senses we will have four combinations. Sense1 of "መግቢያ" has no common word with both senses of "ፈተና". Sense2 of "መግቢያ" has"ክፍል", and "ትምህርት" in common with sense1 of "ፈተና"; and has no common words with sense 2 of "ፈተና". So the lesk weight of sense1 would be the sum of the results from the two combinations which is 5 and sense2 is 0.

$L(s1_{ፈተና}, s_{ትምህርት}) = 3$ $L(s2_{ፈተና}, s_{ትምህርት}) = 0$

$L(s1_{ፈተና}, s1_{መግቢያ}) = 0$ $L(s2_{ፈተና}, s1_{መግቢያ}) = 0$

$L(s1_{ፈተና}, s2_{መግቢያ}) = 2$ $L(s2_{ፈተና}, s2_{መግቢያ}) = 0$

Hence, sense 1 of the target word has a total of 5 common words with the collocated words while sense 2 has no overlap the algorithm assign sense 1 ("ፍተሻ", "ጥያቅ") for the word "ፈተና" in the given sentence.

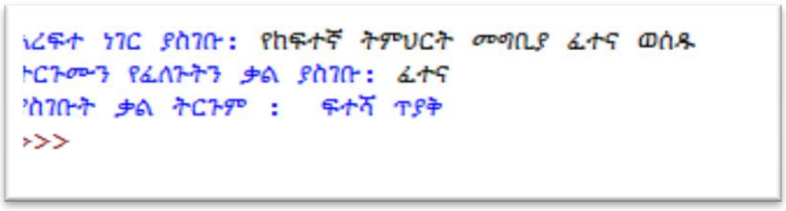


Figure 5.3: Example for Lesk algorithm

Summary of the performance of the Lesk algorithm for the selected words is presented in the table 4.4 below. The algorithm resulted total average of 43.65% accuracy with the highest accuracy of 65% and lowest accuracy of 30%. As indicated in table 4.4 the Lesk algorithm

performed with an average accuracy of 50.1% for nouns, 47% for verbs, 40% for adjectives and 37.5% for adverbs. The performance of the algorithm is highly dependent on the content of the Word-Net. Because few words are included in the Word-Net the performance of the Lesk algorithm resulted average accuracy less than 50%.

Accuracy of Lesk algorithm			
Word classes	**Words**	**Accuracy**	**Average accuracy**
Nouns	ፈተና	61%	50.1%
	እድገት	40%	
	መንገድ	65%	
	ዘር	45%	
	አካል	33%	
	ዋጋ	50%	
	ልጅ	50%	
	ድካም	57%	
	ድምፅ	50%	
Verbs	ደረሰ	44%	47%
	ቆረጠ	50%	
	አከበረ	47%	
Adjectives	ሃሰተኛ	50%	40%
	ደማቅ	30%	
	ትኩስ	41%	
Adverbs	በቀስታ	40%	37.5%
	ወደፊት	35%	
		Total average accuracy	43.65%

Table 5.4: Accuracy of Lesk algorithm

5.2.4. Experiment 4: Applying combination of Cosine similarity with Lesk

In the forth experiment, the combination of Lesk algorithm with the cosine similarity is employed for WSD. In this approach the weight of a sense is identified by adding result from the lesk algorithm with the result from the cosine similarity measure for a given query. And the document with the highest weight will be considered as the sense of the ambiguous word. Given a phrase "ፈተና ውጤት ተቀበለ" to disambiguate a word "ፈተና" using the corpust based lesk algorithm will be as follows: We have already seen the Word-Net entries for the word "ፈተና" above when the Lesk algorithm is discussed. So, here lexical entries of "ውጤት" is presented:

The word "ውጤት" also has two senses:

- sense1(ጥሩ ነገር ፍሬ)፡አንድ ምክንያት ወይም ድርጊት ተከትሎ የመጣ የታየ ሁኔታ ወይም ነገር

- sense2(የፈተና ነጥብ)፡አንድ ሰው ትምህርት ቤት ክፍል በተሰጠው ፈተና ያገኘው ነጥብ ወይም የመለሰው ጥያቄ ብዛት

And there is no lexical entry for the word "ተቀበለ" on the Word-Net. So the system considers only the two words that has dictionary definitions in the Word-Net and counts the overlap between the gloss definition of the two senses of the word "ፈተና" and "ውጤት". The system ignores stop words from the gloss definitions since stop words has less significance to discriminate, so the word the "ወይም"will be discarded. Therefore the number of overlap between sense1 of the word "ፈተና" and sense1 of "ውጤት" is **2 (**"አንድ" and "ነገር"**)** and overlap between sense2 of the word "ፈተና" and sense1 of "ውጤት" is 6 which are**("አንድ",**"ሰው","ትምህርት",", ቤት","ክፍል **","** ጥያቄ") . The third combination which is the first sense of word "ፈተና" and the second sense of "ውጤት" gives one over lap and the forth combination results 0 overlap.

L ($s1_{ፈተና}$, $s1_{ውጤት}$)=2 L ($s1_{ፈተና}$, $s2_{ውጤት}$)=6

L($s2_{ፈተና}$, $s2_{ውጤት}$)= 0 L ($s2_{ፈተና}$, $s1_{ውጤት}$) = 1

As we can see from the result above, we have two lesk weight for each senses of the ambiguous word (for sense1 2 and 6; for sense2 0 and 1). But the highest ones which means weight of sense1 "**6**" and for sense2 "**1**" are taken. From the cosine similarity we have got the following weight:

cos($s1_{ፈተና}$, q)=0.443485287111

cos($s2_{ፈተና}$,q)=0.323396021655

And then the results from the lesk algorithm will be added to the weight of the sense from cosine similarity.

C(ፍተሻ ጥያቅ**)** =0.443485287111 + log 6 **C**(ችግር መከራ ስቃይ)=0.323396021655+ log 1

=1.22160405 =0.323396021655

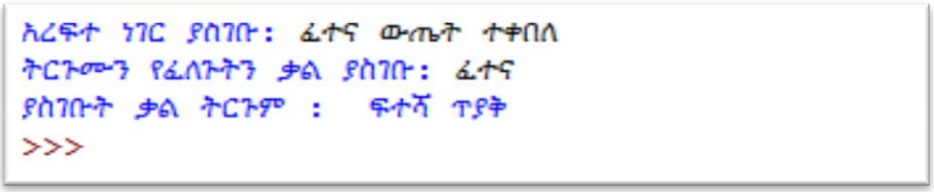


Figure 5.4: Example of WSD using combination of Cosine with Lesk

Applying cosine similarity combined with lesk algorithm resulted total average accuracy of 86.69%. This similarity produced accuracy of 84.1% for nouns, 90.67% for verbs, 92% for adjectives and 80% for adverbs. In this approch we have got accuracy in the range of 60% - 100%. Here again the system performs less for the adverb word classes and resulted the least accuracy for the word "ልጅ" which is 60% and resulted highest accuracy for the words "ዘር" and "ቆረጠ". Like that of cosine similarity, the approach come up with the highest accuracy for the adjecitve word class.

From the result we can see that the cosine similarity measure combined with lesk algorithm produced better accuracy as compared with applying cosine similarity alone. However, the algorithm performed less for the word "ፈተና". The approch improved the performance in 2.17%.

Accuracy of cosine similarity measure with lesk			
Word classes	**Words**	**Accuracy**	**Average accuracy**
Nouns	ፈተና	81%	84.1%
	እድገት	90%	
	መንገድ	95%	
	ዘር	100%	
	አካል	80%	
	ዋጋ	85%	
	ልጅ	60%	
	ድካም	73%	
	ድምፅ	95%	
Verbs	ደረሰ	88%	90.67%
	ቆረጠ	100%	
	አከበረ	84%	
Adjectives	ሃሰተኛ	92%	92%
	ደማቅ	94%	
	ትኩስ	90%	
Adverbs	በቀስታ	70%	80%
	ወደፊት	90%	
		Total average accuracy	86.69%

Table 5.5: Summary of accuracy of cosine with lesk for WSD

5.2.5. Experiment 5: Applying combination of Jacard coefficient similarity with Lesk

The fourth experiment is carried out to test the performance of applying lesk algorithm with Jaccard coefficient similarity for WSD.The combination of Jaccard coefficient with lesk algorithm is the same way as that of combining the cosine similarity. For example the sense of "ፈተና" in the sentence "የእንግሊዘኛ ትምህርት ፈተና ተፈተኑ" can be identified by using this approach as follows:

There is Word-Net entry for the words "ትምህርት" and "ፈተና". Word-Net entries for both words have been shown already above while discussing the Lesk algorithm. The word "ትምህርት" has "ትምህርት", "ቤት", and "እውቀት" in common with sense 1 of "ፈተና" and has no common word with sense 2 of "ፈተና". Then this weight will added to the weight of the senses from the jaccard coefficient.

C(ችግር መከራ ስቃይ) =0.0+(-1)= -1

C(ፍተሻ ጥያቄ) = $0.00974022+\log 3$

= 0.4868614

Since the weight of sense1 is greater than sense 2 the algorithm will return "ፍተሻ ጥያቄ " as the sense of the ambiguous word "ፈተና" in the given context.

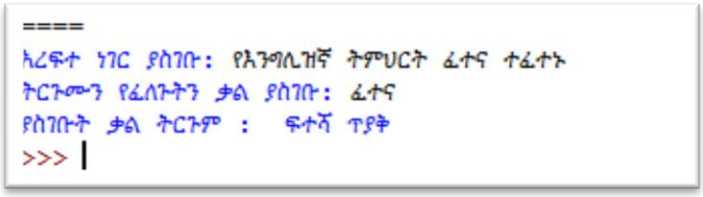


Figure 5.5: Example of WSD using combination of Jaccard with Lesk

Jaccard coefficient combined with Lesk algorithm come up with a total average accuracy of 89.83%, which is 2.65% greater than the accuracy of Jaccard coeffiecient alone. This approach resulted 85% accuracy for nouns, 92.33% accuracy for verbs, 92% accuracy for adjectives and 90% accuracy for adverbs. Like other approachs this approach performs less for nouns. Come up with the highest accuracy for adjective word class. The word "ቆረጠ" disambiguated with 100% accuracy. As that of WSD based on Jaccard similarity, this approach also produced highest

accuracy for verbs. And the WSD based on Jaccard similarity combined with Lesk give the highest accuracy.

Accuracy of Jaccard coefficient similarity with Lesk			
Word classes	**Words**	**Accuracy**	**Average accuracy**
Nouns	ፈተና	86%	85%
	እድገት	90%	
	መንገድ	90%	
	ዘር	95%	
	አካል	80%	
	ዋጋ	85%	
	ልጅ	60%	
	ድካም	84%	
	ድምፅ	95%	
Verbs	ደረሰ	88%	92.33%
	ቆረጠ	100%	
	አከበረ	89%	
Adjectives	ሃሰተኛ	92%	92.0%
	ደማቅ	94%	
	ትኩስ	90%	
Adverbs	በቀስታ	80%	90%
	ወደፊት	100%	
		Total average accuracy	89.83%

Table 5.6: Summaryof Accuracy Jaccard coefficient similarity with Lesk for WSD

5.3. Findings and challenges

This thesis demonstrated the implementation of Lesk algorithm with annotated corpus for Amharic WSD. For this, WSD system is developed which is composed of tagged example sentences and Word-Net. Information from the tagged example is extracted by using similarity measures; and from the Word-Net Lesk algorithm is employed. So, the end result of this study illustrates the performance of combining this two information sources for identifying the correct sense of a target word based on a given context.

A corpus based Lesk algorithm disabmiguate one target word combining example sentences and dictionary definitions. However, in this study before combining the two, the performance of the WSD based on each knowledge sources are tested individually. The performance of the Amharic WSD based on example sentences alone have been tested by applying two similarity measures: Cosine similarity and Jaccard coefficient. And to test the performance of WSD based on dictionary definition simple Lesk algorithm alone is utilized.

Experimental results shows that the Amharic WSD developed based on cosine similarity, Jaccard coefficient, Lesk algorithm, cosine combined with Lesk and Jaccard coefficient with Lesk resulted an average accuracy of 84.52%, 85.96%, 43.65%, 86.77% and 89.83% respectively. The comparison interms of average accuracy scored by the semantics measures is summarized below in table 5.6.

Summary of Average accuracy					
Word classes	Cosine	Jaccard	Lesk	Cosine Lesk	Jaccard Lesk
Nouns	81.94%	83.68%	50%	84.1%	**85%**
Verbs	88.67%	90.67%	47%	90.67%	**92.33%**
Adjectives	90%	87%	40%	**92.33%**	92.0%
Adverbs	77.5%	82.5%	37.5%	80%	**90%**
Average Accuracy	84.52%	85.96%	43.65%	86.77%	**89.83%**

Table 5.7: Summary of Average accuracy of the four methods

The system developed has been tested with 17 words that are taken from noun, verb, adjective and adverb word classes. In all of the experiments the system performs with lowest accuracy for adverbs. This is because, most of the adverbs appear with similar words in the example sentences of the different senses. Cosine with or without Lesk resulted higher accuracy for adjectives and Jaccard coefficient with or without Lesk resulted higher accuracy for verbs.

As shown in the above table 5.6, Jaccard coefficient similarity performs better than Cosine similarity except in case of adjectives. The two experments evidenced that similarity measures can be used for distinguishing the sense of a target word in a given context. The selected similarity measures disambiguated the chosen words with average accuracy of more than 85%. This indicates that the two similarity measures promising performance if we employee them using a tagged corpus. The Jaccard Coefficient performed 1.44% better than Cosine similarity for WSD.

The performance Lesk algorithm alone is also tested for Amharic WSD. The Lesk algorithm performed less for every word and come up with average accuracy of 43.65%. However, as depicted in table 5.6, both of the similarity measures performs better when combined with Lesk algorithm. The Cosine similarity combined with Lesk algorithm performs 2.17% better than

Cosine similarity and Jaccard combined with Lesk performs 3.87% better than Jaccard coefficient similarity. The Lesk algorithm uses dictionary definitions and semantic similarity between the target word and the collocated words. So combining this information with the information from the example sentence helps the system to identify the sense of a target word.

The major challenge in conducting this research was organizing example sentences and the Word-Net. The main sources of information for the WSD system developed are Word-Net and example sentences, however there is no well organized Word-Net and corpus for Amharic language.Therefore, we have tried to construct a simple Word-Net and organize corpus. As stated earlier the Word-Net constructed by Samrawit is used by adding some words. The Word-Net is composed of words with their synonymy and gloss definition. Two dictionaries are used for adding on Samrawit [**35**] Word-Net, on those dictionaries for some of the words there is no synonym list and for some of them there is no gloss definition. So it was difficult task to organize words with their synonym and gloss definitions.

As a result, very small number of words are included on the Word-Net and the lesk algorithm works only for those words included on the Word-Net. Hence, we get less performance using Lesk algorithm and little improvement combining lesk algorithm with the similarity measures. The other limitation of the Lesk algorithm is that, it strictly considers the definition of a synonym as a source of contextual information for a given sense. Employing adapted Lesk algorithm that takes in account related concepts and their definitions would be helpful to increase the accuracy. According to report of Senseval-2, the adapted Lesk algorithm doubled the accuracy of the WSD. This algorithm is not employed for this study because of lack of dictionaries that incorporate hypernyms, hyponyms, troponyms and their associated definitions.

The attempt of the study was to develop a WSD system that will work with any text from different domain. Therefore, it was expected to collect example sentences from different sources. Even if texts in Amharic language are more available than any other local language on WWW, it is still very challenging, difficult and time consuming to collect and organize example sentences for each senses of the words. In addition, for some of the words one sense has dominant frequency of occurrence than other senses and some senses occur very rarely. This makes it so hard to find equal number of example sentences for every senses of a word. The other

challenging thing while organizing the corpus was to manually tag the senses, because sometimes it is so confusing and difficult to differentiate the correct sense of the words in the example sentences. And some of the words have senses that are difficult to understand and differentiate.

Consequently, only 100 sentences are collected for each sense of the words. The accuracy of the system based on both similarity measures is dependent on the frequency of words that occur with the senses of the target word in the example sentences. The system identifies the sense of a target word by comparing the similarity between the documents contianing example sentences of the target word and the input text. So the system works only if the input sentence includes words that are contained in the 100 example.

In some cases, words that are collocated with the target word in the input text may appear with nearly equal frequency in the example sentences of each senses of the target word. In such cases the system become confused to identify which sense of the word occurred in that particular context. So, the system come up with a least accuracy for those words that are frequently collocated with similar words in their different senses.

In this study a more generic approach was explored for Amharic WSD which can disambiguate any ambiguous word. However, since the system developed works given a tagged example sentence and tagged corpus was collected for only 17 words, the system can not identify the ambigous word and can not disambiguate all ambiguous words in the input text. So, in the experiments the system is tested for disambiguating a given targetword selected by the user from an input text.

Summary

This chapter presented the experiments conducted and results obtained from the experiments. Five experiments were conducted using 10 test examples for each sense of the selected 17 ambiguous words. The experiments was done on Cosine similarity measure, Jaccard coefficient, Lesk algorithm, combination of Cosine with Lesk and Jaccard with Lesk and resulted average 84.52%, 85.96%, 43.65%, 86.69% and 89.83% respectively. Jaccard combined with Lesk come up with the highest accuracy. And the system come up with a least accuracy for those words that are frequently collocated with similar words in their different senses.

CHAPTER SIX

Conclusion And Recommendation

6.1. Conclusion

The main aspiration of this research work is to investigate a more generic approach for all-words based Amharic WSD. Previous thesis works in the area of WSD conducted a study on lexical sample task and their experiments was on verbs using machine learing approach. In this research a new methode is explored towards all words task by using Lesk with annotated corpus. To this end, a WSD system developed based on implementation of Lesk algorithm with similarity measures.

WSD system that disambiguate an ambiguous word based on context is implemented using python programming language. The system uses mannually tagged example sentences and Word-Net as a knowledge source. And to identify the sense of the word the system measures similarity between the input sentence and the documents containing example sentences of the different senses of the target word. In addition, the system employees Lesk algorithm. To measure similarity, Cosine similarity measures and Jaccard Coefficient are selected. The WSD system is tested using 17 Amharic ambiguous words which are selected randomly from different domain. The selected words are from noun,verb, adjective and adverb word classes.

In this study, a new method is designed for developing Amharic WSD that uses similarity measures to extract information from corpus and Lesk algorithm from Word-Net to disambiguate words from different classes. The performance of the WSD based on Cosine similarity and Jaccard Coefficient alone achieved 84.52% and 85.96% accuracy respectively. For most of the cases the two similarity measures have equal performance but as we can see from the total average accuracy the Jaccard performed a little bit better than the cosine. From this we can conclude that both the similarity measures can be employeed for WSD.

The Lesk algorithm alone performed with total accuracy of 43.65% which is less than the performance of the similarity measures. The less performance of the Lesk algorithm is because, the Word-Net incorporate small number of words. And the average accuracy of performance Cosine similarity with Lesk resulted 86.69% and Jaccard Coefficient with Lesk resulted 89.83% total. On the average, the Amharic WSD performs better when Jaccard coefficient combined

with Lesk algorithm with total average accuracy of 89.83%. The result showed that the similarity measures performed better when they are combined with Lesk algorithm and from this we can conclude that it is better to use the similarity measures combined with Lesk to come up with enhanced result.

In general, this study shows that given tagged example sentences and Word-Net entries for a given ambiguous word from any Amharic word classes; we can use this system to identify the sense of the word by measuring the similarity between input sentence in which the ambiguous word is in and the Lesk algorithm.

6.2. Recommendation

Based on the findings of the study, the following recommendations are formulated as a future research direction in the area of Amharic WSD:

- The Amharic WSD developed identifies word senses using information from Word-Net. We have tried to constuct a simple Word-Net that incorporate only words with their synonyms and gloss definitions . And the Word-Net is not as much well-structured and contains small number of words which affects greatly the performance of the Amharic WSD. Hence we recommend to work more on constructing an Amharic Word-Net.

- Even though, there are different types of sense relationships between words that will help to identify the sense of a word, the Word-Net constructed incorporate only synonmous relationship. For the future it would be better if the Word-Net incorporate other word sense relationships like Hyponymy.

- The other major soure of information for the WSD system developed in this thesis is tagged example sentences. We have collected and organized only 100 sentences for each senses of selected 17 Amharic ambigous words. It is obvious that as the number of examples increases the accuracy of the system will increase becuase when we use more sentences we can get more words that are related to the target word. Therefore, we recommend that corpus containing more words and more tagged example sentences would be organized.
- We have used a simple Lesk algorithm, which identifies word senses by finding the sense that leads to the highest overlap between its dictionary definition and the current context.

There are other variation of simple Lesk algorithms like Adaptive Lesk algorithm which Access a dictionary with senses arranged in a hierarchical order.This extended version uses not only the gloss/definition of the synset but also considers the meaning of related words. We suggest for researchers who will work on the field of WSD using this method to employee Adaptive Lesk algorithm instead of simple Lesk algorithm and check the performance.

- The WSD protoytpe developed in this research works for a single word in the given text and could not identify the ambiguous word from the given text. In the future it is better to incorporate a way to identify the ambiguous words in the given text and disambiguate all the ambiguous words in that given text.
- Since WSD is a central problem in the field of NLP, different attempts are being done in this area for other local languages using different approaches. For the future we suggest for researchers who are interested to work on WSD for other local language to follow this approach and to explore more.

References

[1] Alok Ranjan Pall and Diganta Saha2, "Word Sense Disambiguation: A Survey," *IJCTCM*, vol. 5, no. 3, pp. 1-16, July 2015.

[2] Andres M., Armando S., German R., Manuel P., "Combining Knowledge and Corpus-based Word Sense Disambiguation Methods," *Journal of Artificial Intelligence Research*, vol. 23, no. 1, pp. 299-330, January 2005.

[3] Navigli Robert, "Word Sense Disambiguation: A Survey," *ACM Computing Surveys*, vol. 41, no. 2, pp. 1-69, February 2009.

[4] Nancy Ide and Jean Veronis , "Word Sense disambiguation: The state of the Art," *Computational Linguistics*, vol. 24, no. 1, pp. 2-40, March 1998.

[5] William Collins Sons, ""Dictionary.com". ," Available:WWW.dictionary.com/browse/amharic Retrieved 2016-09-13, 2012.

[6] Samia Zekaria , "Ethiopian Population Census," Central Statistical Agency, A.A, Statistical report 2007.

[7] " Amharic Explained," Available: http://everything.explained.today/Amharic/ Retrieved: 2016-09-13,.

[8] Solomon Mekonen , "Word Sense Disambiguation for Amharic Text: A machine Learning Approach ," AAU, Addis Ababa, Masters thesis 2010.

[9] Solomon Assemu , "Unsupervised Machine Learning Approach for Word Sense Disambiguation to Amharic," AAU, Addis ababa, masters thesis 2011.

[10] Getahun Wassie , "A word sense Disambiguation model for Amharic Words using Semi supervised learning paradigm," AAU, Addis Ababa, Masters thesis 2014.

[11] Gashaw Wudu , "Unsupervised Corpus based Approach to Amharic Word Sense Disambiguation," Gonder University, Gonder, Masters Thesis 2015.

[12] Baye yemam , *የአማረኛ ሰዋሰው*. A.A, Ethiopia, 1986 ዓ.ም.

[13] Getahun Amare , "Towards the Analysis of Ambiguity in Amharic," *Institute of Ethiopian Studies*, vol. 34, no. 2, pp. 35-56, December 2001.

[14] Kumar Singh Yogesh, *Fundamental of Research Methodology and Statistics*. Mumbai, India: New Age International, 2006.

[15] R. Kothari C., *Research Methodology: Methods and Techniques*, 2nd ed. Jaipur, India: New Age International Publisher, 2004.

[16] Ethiopian Linguistic and research institute, *አማርኛ መዝገበ ቃላት*. Addis Ababa, Ethiopia: Artistic publisher, 1993.

[17] "The python Tutorial--python documentation.html ,".

[18] Eneko Agirre and Mark Stevenson , *Word Sense Disambiguation: Algorithms and Applications*, Eneko Agirre and Philip Edmonds , Eds.: Springer, 2007.

[19] Gerald Kowalski , *Information Retrieval Systems: Theory and Implementation*, 3rd ed., Bruce Croft W., Ed. Boston, United states of America: Kluwer acadamic publishers, 1999.

[20] Manning Christopher and Hinrich Schutze , *Foundations of Statistical Natural Language Processing*, 2nd ed. London, England: The MIT press, 1990.

[21] Gerard Escudero Bakx, *Machine Learning Techniques For Word Sense Disambiguation*. Barcelona, Spain, 2006.

[22] David Maartinez , Eneko Agirre , and Xinglong Wang , "Word Relatives in context for Word Sense Disambiguation," in *Australasian Language Technology Workshop*, Melbourne, 2006, pp. 42-50.

[23] "Wordnets in the world," Available:globalwordnet.org/wordnets-in-the-world Retrieved:5/7/2016,.

[24] J. Sreedhar , S. Viswanadha , A. Vinaya , and P. Kumar , "Word Sense Disambiguation: An Empirical Survey," *IJSCE*, vol. 2, no. 2, pp. 494-503, May 2012.

[25] Daniel Jurafsky and James Martin , *Speech and Language processing: An introduction to natural languge processing*., 2007.

[26] Yogita Rani and Harish Rohil , "Comparative Analysis of BIRCH and CURE Hierarchical Clustering Algorithm using WEKA 3.6.9," *CSEA*, vol. 2, no. 1, pp. 25-29, January-February 2014.

[27] David Martinez , Eneko Agirre , and Xinglong Wang , "Word Relatives in Context for Word Sense Disambiguation," in *Australasian Language Technology Workshop*, Australia, 2006, pp. 42-50.

[28] F. Zopon Boshra and Shashank Joshi , "Comparative Analysis between Naive Bayes Algorithm and Decision Tree to solve WSD Empirical approach," *Lecture Notes on Software Engineering*, vol. 4, no. 1, pp. 82-86, February 2016.

[29] Xiaohua Zhou and Hyoil Han , "Survey of Word Sense Disambiguation Approaches," in *FLAIRS Conference*, Philadelphia, 2005.

[30] Yihua Chen and Maya R. Gupta , "EM Demystified: An Expectation-Maximization Tutorial," University of Washington, Washington, Technical Report 2010-0002, 2010.

[31] Sudipto Guha , Rajeev Rastogi , and Kyuseok Shim , "CURE: An Efficient Clustering Algorithm for Large Databases," in *ACM*, Washington, 1998, pp. 73-84.

[32] Tian Zhang , Raghu Ramakrishnan , and Miron Livny , "BIRCH: An Efficient Data Clustering Method for Very Large," in *ACM*, Montreal, 1996, pp. 103-114.

[33] PHILIP EDMONDS and ADAM KILGARRIFF , "Introduction to the special issue on evaluating word sense disambiguation systems," *Natural Language Engineering*, vol. 8, no. 4, pp. 279-291, August 2002.

[34] FEYISA GEMECHU , "Unsupervised Corpus Based Approach for Word Sense Disambiguation to Afaan Oromo Words," Addis Ababa, 2015.

[35] Samrawit Zewdneh , "Word Sense Disambiguation using Semantic Similarity for Query Expansion in Amharic Information Retrieval," AAU, Addis Ababa, Thesis report 2014.

[36] Amanda Wimsatt and Rachel Wynn , "National Language of Ethiopia," Texas state university , Texas, Amharic Language and Culture Manual 2011.

[37] C.H. Dawkins , *Fundamentals of Amharic*. Addis Abaaba, Ethiopia: Sudan interior mission, 1960.

[38] Million Meshesha and C. V. Jawahar , "Indigenous Scripts of African Languages," Institute of Information Technology,.

[39] Charles Wiliam Isenberg, *Grammer of the Amharic Language*, 1st ed. London, England: Stanford University, 1985.

[40] Aliwy H. and Abbas R. , "IMPROVEMENT WSD DICTIONARY USING ANNOTATED CORPUS AND TESTING IT WITH SIMPLIFIED LESK ALGORITHM," , Baghdad, 2015.

[41] Zergaw Gezaw , "ዘርጋው የአማርኛ መዝገበ ቃላት," AAU, Addis Ababa, 1994.

[42] Amanuel Hirpa Madessa, "Probablistic information retrieval system," AAU, Addis Ababa, Masters thesis 2012.

[43] Gerald Kowalski, *Information Retrieval Systems Thoery and Implementation*. Boston: Kluwer Acadamic publishers, 1997.

[44] Manning Christopher D. , Prabhakar Raghavan , and Hinrich Schütze , *An Introduction to Information Retrieval*. Cambridge, USA: Cambridge University Press, 2009.

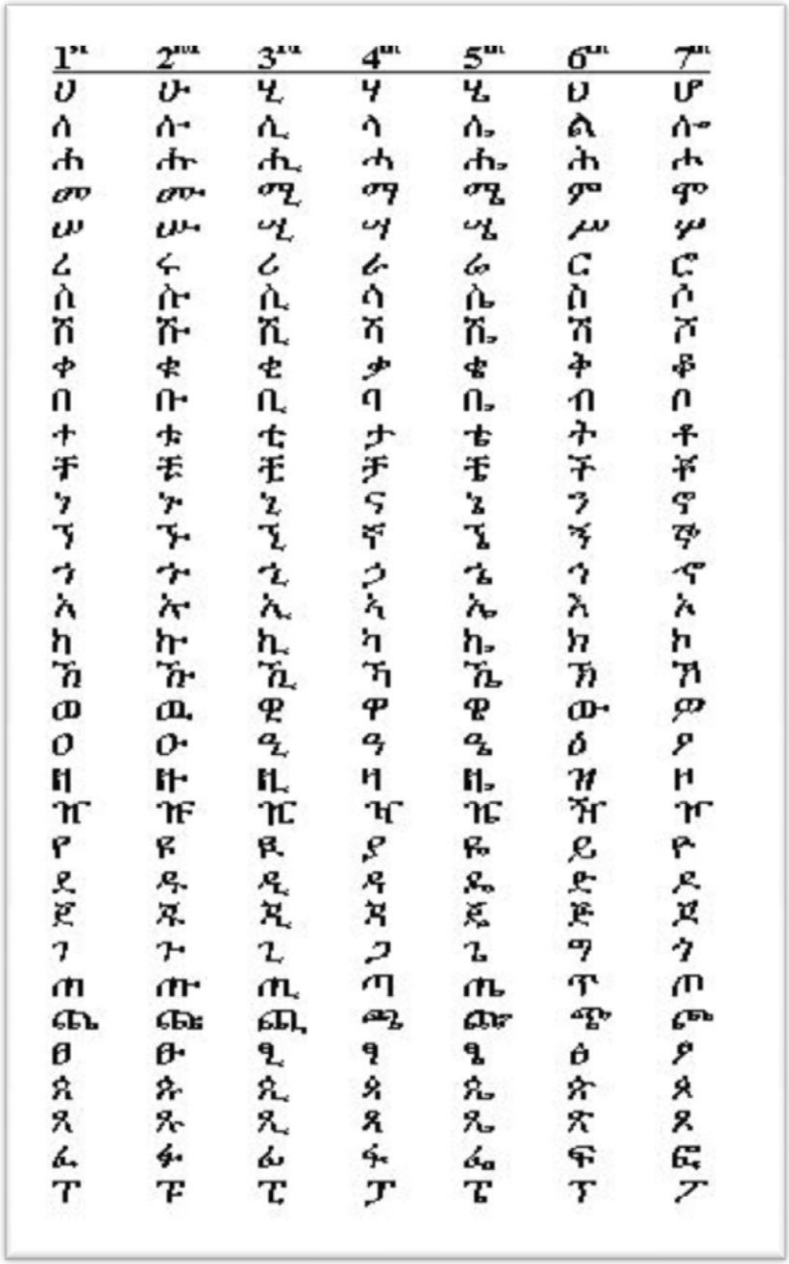
[45] Neha Agarwal , Mukesh Rawat , and Vijay Maheshwari , "Comparative analysis of Jaccard coefficient and cosine similarity for WEB document similarity measure," *International Journal for Advance research in engineering and technology*, vol. 2, no. X, pp. 18-21, Oct 2014.

[46] Zeynep ALTAN , "WORD SENSE DISAMBIGUATION: METHODS AND APPLICATIONS," , İSTANBUL.

[47] Getahun Wassie , "A word sense Disambiguation model for Amharic Words using Semi supervised learning paradigm," *Technology and Arts Research Journal*, vol. 3, 2014.

APPENDIXES

Appendix 1: Amharic Alphabets



1ኛ	2ኛ	3ኛ	4ኛ	5ኛ	6ኛ	7ኛ
ሀ	ሁ	ሂ	ሃ	ሄ	ህ	ሆ
ለ	ሉ	ሊ	ላ	ሌ	ል	ሎ
ሐ	ሑ	ሒ	ሓ	ሔ	ሕ	ሖ
መ	ሙ	ሚ	ማ	ሜ	ም	ሞ
ሠ	ሡ	ሢ	ሣ	ሤ	ሥ	ሦ
ረ	ሩ	ሪ	ራ	ሬ	ር	ሮ
ሰ	ሱ	ሲ	ሳ	ሴ	ስ	ሶ
ሸ	ሹ	ሺ	ሻ	ሼ	ሽ	ሾ
ቀ	ቁ	ቂ	ቃ	ቄ	ቅ	ቆ
በ	ቡ	ቢ	ባ	ቤ	ብ	ቦ
ተ	ቱ	ቲ	ታ	ቴ	ት	ቶ
ቸ	ቹ	ቺ	ቻ	ቼ	ች	ቾ
ነ	ኑ	ኒ	ና	ኔ	ን	ኖ
ኘ	ኙ	ኚ	ኛ	ኜ	ኝ	ኞ
ኀ	ኁ	ኂ	ኃ	ኄ	ኅ	ኆ
አ	ኡ	ኢ	ኣ	ኤ	እ	ኦ
ከ	ኩ	ኪ	ካ	ኬ	ክ	ኮ
ኸ	ኹ	ኺ	ኻ	ኼ	ኽ	ኾ
ወ	ዉ	ዊ	ዋ	ዌ	ው	ዎ
ዐ	ዑ	ዒ	ዓ	ዔ	ዕ	ዖ
ዘ	ዙ	ዚ	ዛ	ዜ	ዝ	ዞ
ዠ	ዡ	ዢ	ዣ	ዤ	ዥ	ዦ
የ	ዩ	ዪ	ያ	ዬ	ይ	ዮ
ደ	ዱ	ዲ	ዳ	ዴ	ድ	ዶ
ጀ	ጁ	ጂ	ጃ	ጄ	ጅ	ጆ
ገ	ጉ	ጊ	ጋ	ጌ	ግ	ጎ
ጠ	ጡ	ጢ	ጣ	ጤ	ጥ	ጦ
ጨ	ጩ	ጪ	ጫ	ጬ	ጭ	ጮ
ፀ	ፁ	ፂ	ፃ	ፄ	ፅ	ፆ
ጸ	ጹ	ጺ	ጻ	ጼ	ጽ	ጾ
ፈ	ፉ	ፊ	ፋ	ፌ	ፍ	ፎ
ፐ	ፑ	ፒ	ፓ	ፔ	ፕ	ፖ

Appendix 2: Amharic Word-Net

ድምጽ@ጩህት:በመናገር በማዜም በእንቅስቃሴ አማካይነት የሚፈጠር ጆሮ ሊሰማ የሚችል ስርአት የተከተለ ጩህት;ውሳን:አንድን ጉዳይ በመደገፍ ወይም በመቃወም የራስን ውሳኔ ምርጫ በይፋ ወይም በሚስጢር ማሳወቅ

ጸሀይ@ኮከብ ጀምበር:መሬትና ሌሎች ፕላኔቶች ዙራዋን የሚዞር ብርሃን ሙቀት የምትሰጥ

ሃሰተኛ@ውሸት የሚናገር:ሁልጊዜ የሚዋሽ ሀሰት የሚናገር አስመሳይ ሰው;ትክክልኛ ያልሆነ: እውነተኛ ያልሆነ ገንዘብ ኖት ሰነድ መረጃ

መታወቂያ@መለያ ካርድ:ለቀበሌ ነዋሪዎች ሰራተኞች መለያ እንዲሆን ፎቶና አስፈላጊው መረጃ ተጽፎበት የሚሰጥ ካርድ ወይም ወረቀት

ሰነድ@መዝገብ:ስለአንድ ጉዳይ መረጃ የያዘ ጽሁፍ ማስረጃ

ወሬ@ዜና:አዲስ የሚሰማ ነገር;ጭውውት:በሰዎች መካከል የሚደረግ ጭውውት ንግግር የሃሳብ ልውውጥ

ቃል@የድምጽ ምልክት:አንድን ጽንሰ ሃሳብ ለመወከል ድምጽ የተነገረ ወይም በጽሁፍ የሰፈረ ምልክት አነስተኛ የአረፍተ ነገር ክፍልፋይ;ሃሳብ:ስለ አንዳንድ ነገር የታየ ተነገረ የተጻፈ ሃሳብ አስተያየት መልእክት

ነቢያት@የወደፊቱን ተናጋሪ:እውነት ሆነ ሀሰት ወደ ፊት ሊሆን ሊደርስ የሚችለውን ነገር አስቀድሞ የሚናገር ሰው

አስተማሪ@መምህር:በማስተማር በማሳወቅ ሙያ ላይ የተሰማራ ሰው

ምስክር@እማኝ:ስለአየው ስለሰማው ነገር ህጋዊ በሆነ ቦታ ቀርቦ በመሃላ ቃሉን የሚሰጥ ሰው

ምላስ@ምላስ:ለመናገር ለመቅመስ ለመላስ የሚያገለግል በአፍ ውስጥ የሚገኝ የሰውነት አካል

ሰው@ሰብአዊፍጡር:ነብስ ና ስጋ ያለው ማሰብ ማሰላሰል የሚችል ፍጡር

አከበረ@ከፍ አደረገ:ወግ ማእረግ ክብር ሰጠ;በአል ማክበር:በአል ዝግጅት ስነስርኣት ላይ ተሳተፈ

ትኩስ@ሙቅ:ገና ከእሳት የወረደ ውሃ ምግብ ቡና ሻይ;አዲስ ለጋ:ለምለም ከተገዛ ወይም ከተሰራ ያልቆየ እቃ ያላረጀ ጉልበት

ዜና@ወሬ:ከሰው አንደበት የሚወወጣ አዲስ ያልቆየ የሚሰማ ነገር

ወጣት@ወጣት ልጅ:እድሜው ከአስራ ስምንት እስከ ሰላሳ አመት የሚሆን ሰው;ጠንካራ ብርቱ:አዲስ ያልደረጀ ገና ለጋ

ደማቅ@የጎላ:ቦግ ያለ ጉልህ ሆኖ የሚታይ ቀለም መልክ ምስል ጸሃይ ብርሃን ወጋገን;የሞቀ:የተጋጋለ ጨዋታ ጭፈራው በአል ገበያ ደራ ሞቀ

ቀለም@አረንጉዴ ቀይ ሰማያዊ ነጭ ጥቁር:ልዩ ልዩ መልክ ያለው የተቀመመ ለመጻፍ ለመሳል ለመቀባት የሚያገለግል ነገር

ኢዮቤልዩ@የሃያ አምስት አመት በአል:አንድን ትልቅ ድርጊት ወይም ሁኔታ ለማስታወስ በየሃያ አመት የሚከበር በአል

ቆረጠ@ገመደ:አንድ ነገር ጎረደ በጠሰ;ወሰነ:አንድ ነገር ለማድረግ ጨከነ ቀን ዋጋ ወሰነ

ደረሰ@ተፈጸመ ሆነ:አንድ ነገር ሲሆን ሲከሰት ሲያጋጥም ሲከናወን;በቃ:ለተለያየ የእድገት የጥራት ደረጃ ላይ መድረስ

መከራ@ፈተና ጭንቅ ስቃይ:በአካል በአእምሮ ላይ ጭንቅ ችግር ህመም ጉስቁልና ሲያጋጥም

አካል@ገላ መላ ሰውነት:እያንዳንዱ ሰውነት ክፍል;ማህበር ጭፍራ:አንድን ተግባር የሚፈፅም ቡድን አባል;ክፍል:የሆነ የአንድ ነገር ክፍል

እድገት@የአካል ክፍል መጨመር: የአጥንት የአእምሮ የጡንቻ ቁመት ወይም ሌላ ሰውነት ክፍል ማደግ;የተሻለ ማግኘት:መበልፀግ ሃብት ኢኮኖሚ ልማት ስራ ከፍ ላቅ ያለ ደረጃ መድረስ

ዘር@ወገን ዘመድ ትውልድ:የትውልድ ሀረግ የሚታወቅበት በደም ግንኙነት የተሳሰረ የዝምድና ሰንሰለት ልጅ ተወላጅ;ሲዘሩት የሚበቅል ፍር:አስፈላጊ ነገሮች ሲሞሉለ (አፈር ውሃ አየር ሙቀት) የሚበቅል

ሃይል@ብርታት ጥንካሬ ጉልበት:አንድን ተግባር ለማከናወን የሚያስችል ጉልበት አቅም;ቡድን:አንድን ተግባር የሚፈፅም ቡድን;አስገዳጅ:ዘዴ ያልተከተለ አስገዳጅ ግፊት

መንገድ@ብልሃት ዘድ አሰራራ:የሆነ ስራለመስራት የሚጠቅም እውቀት ጥበብ;ንዳና:መሄጃ መመላለሻ መተላለፊያ ቦታ

ልጅ@ህጻን:በቅርብ ጊዜ የተወለደ በእድሜ ያልገፋ;የአብራክ ክፋይ:ከአብራክ የወጣ

ፈተና@ፍተሻ ጥያቅ:አንድ ሰው ስለአንድ ያለውን እውቀት ለመመዘን ትምህርት ቤት ክፍል ውስጥ በፅሁፍ በቃል የሚቀርብ ጥያቄ;ችግር መከራ ስቃይ:በልዮ ልዮ ነገር ችግር ስቃይ ሲያጋጥም

ተማሪ@ተማሪ:ትምህርት ቤት ገብቶ በመማር ላይ ያለ ሰው;empty

ውጤት@ጥሩ ነገር ፍሬ:አንድ አንድ ምክንያት ወይም ድርጊት ተከትሎ የመጣ የታየ ሁኔታ ወይም ነገር;የፈተና ነጥብ: አንድ ሰው ትምህርት ቤት ክፍል ውስጥ በተሰጠው ፈተና ያገኘው ነጥብ ወይም የመለሰው ጥያቄ ብዛት

መግቢያ@በር:ወደ ውስጥ የሚያስተላልፍ በር; መጀመሪያ: ወደ ሌላ ክፍል ትምህርት ደረጃ መዘዋወሪያ ፈተና

ወረቀት@ፅህፈት መሳሪያ:ከእንጨት ከልጥ ከገለባ ጨርቅ ከመሳሰሉት ነገሮች የሚሰራ ፅህፈት መሳሪያ;ደብዳቤ:ማዘዛ

ባህል@ወግ ልምድ እምነት:የህብረተሰብ የአናናር ዘዴ ብልሃት ወግ ልምድ እምነት

አእምሮ@ህሊና:ለማሰብለማገናዘብ የሚጠቅም ሰውነት አካል ክፍል

ብልፅግና@ሃብታምነት:ኢኮኖሚ እድገት

ሰውነት@ገላ አካላት:የሰው መላ አካል

አጥንት@የስጋ ምሰሶ ወጋግር:በስጋና በቆዳ የተሸፈነው ጠንካራ የሆነ ሰውነት ክፍል

እረፍት@ማረፍ እፎይ ማለት: ለተወሰነ ጊዜ ስራ ትምህርት ተለይቶ የሚቆይበት ጊዜ

ድካም@አቅምማጣት: በተለያየ ስራ ትምህርት ወይም ህመም ምክንያት የሚከሰት የመዛል ስሜት;ጥረት:የሆነ ነገር ለማግኘት ወይም ግብ ለማሳካት የሚደረግ ጥረት ልፋት

ዋጋ@ተመን:አንድ ነገር የሚገዛበት ወይም የሚሸጥበት አንድ ስራ ለተሰራበት ወይም ለተሰጠበት አገልግሎት የሚከፈል የገንዘብ መጠን ልክ ተመን;ፋይዳ ጥቅም:ለሰዎች የሚሰጡት ጠቀሜታ ያላቸው ጥሩነት ጥቅም

Appendix 3: Example sentences from the corpus

የአንጀት ቁስለት ህመም አልፎ አልፎ ዘር ሊከሰትም ይችላል፡፡

የዘር ሐረጎች ከተራ የትውልድ ሰንሰለትነት የዘለለ ፋይዳ አላቸው የሚለው ነገር ለብዙዎች ግልፅ አይደለም

በያዝነው መኸር ወቅት ከ13 ሚሉዮን ሄ/ርመሬት በላይ በተለያዩ ዘር ለመሸፈን እቅድ ተይዞ እየተተገበረ ይገኛል፡

“ምርጥ ዘር” በመጠቀም ና ከዘሩም ጋር የተለያዩ ሰው ሰራሽ የኬሚካል ማዳበሪያዎችንና መድሀኒቶችን በመጠቀም የአለምን ህዝብ የምግብ ፍላጎት ማርካት ይቻላል

ቼሎ የሚባል አንድ የቅርጻ ቅርጽ ባለሙያ ዝነኛ ከመሆን የበለጠ ከፍተኛ ዋጋ ያለው ነገር እንዳለ ማስተዋል ችሏል።

ሕይወት ከሌለ ማንኛውም ነገር ዋጋ የለውም።

አትክልቶቹ ቶሎ የሚበላሹ ወይም ለአደጋ የሚጋለጥና ጠቃሚ ዋጋ የሚያወጡ ከሆነበ ጊዜ ይሰበስባሉ፤

በትዕዛዝ ለሚሰሩ በእጅ የሚሰራ የሳጥን መጫኛ ና ሳጥኖቹ ዋጋቸው እስከ 750 የአሜሪካን ዶላር ሊያወጡ ይችላሉ፡፡

ምግብ አስተዳደር መመሪያ ተዘጋጅቶ የኤድስ ህሙማን ምግብ የሚያገኙበት መንገድ መቻቸቱን ለአብነት ተናግረዋል

በአጠቃላይ ሥራው ወጥ በሆነ መንገድ እየተሠራ አለመሆኑን በመገንዘብ ሰፊ ሥራ መስራት ይጠይቃል

እያንዳንዱ ልጅ በት/ቤት ያን የመጀመሪያ እርምጃቸውን ቢራቢሮዎች እና ችሎታ፣ እምቅ እና ህልሞች ይዘውይ ጀምሩታል።

በጉዞው ላይ የሚሳተፉ ከሆነ ህጻን ልጅ ይዘው እንዳይመጡ ምክንያቱም ይህ ልዩ ፕሮግራም የተዘጋጀው ለልጆ እና ለክፍል ጓደኞች ብቻ ነው፡፡

አንድ ልጅ የመጀመሪያ ቋንቋ ለእርሱ ወይም ለእርሷ ማንነት ወሳኝ ነው።

ልጅዎ ከአጋጠማቸው አዳዲስ ልምዶች ጋር ሙከራ እንደሚያደርጉ አይነት አድርገው ማነቃቃትና ማበረታታት ነው።

ብዙ ፈተና ቢገጥመውም፣ ባሻገር የተሻለ ነገር እንደሚያገኝ ተስፋአድርጎ ነበር፡፡

አሁን ፈተናውን ለመቋቋም የቻልነው ደግሞ አንድ በመሆናችን ነው፡፡

ክፍል ብሄራዊ ፈተና ተፈታኞች በሰላም ፈተናቸውን ወስደዋል።

ፈተና ተማሪዎች በየደረጃው ያገኙት እውቀት ላይ የደረሱበትን ብቃት መመዘን የሚያስችል ስርአት በመሆኑ የትምህርትን ጥራት በማስጠበቅ ረገድ ቀዳሚ ድርሻ አለው።

አካል ብቃት እንቅስቃሴ ማድረግ ሌላው ጠቀሜታ አዕምሮ በሙሉ አቅሙ እንዲሰራ ማድረግና የፈጠራ ብቃቱን ከፍማድረጉ ላይመሆኑም በዚህ ጥናት ተጠቅሷል።

በፀጥታ ሃይል አባላት ላይ በደረሰው የህይወትና አካል ጉዳት የተሰማቸውን ጥልቅ ሃዘንም ከልሎቹ ገልጸዋል።

የህግ የበላይነት መርህ ውሳኔዎች ህግንና ደንብን የተከተሉና በውሳኔ ሰጪው አካል ስልጣን ስር ያሉ መሆን እንደሚገባቸው ያመለክታል።

ድምጽን ከፍ በማድረግ የተነበቡ የታተሙ ጽሁፎችን የመረዳት ችሎታን ያረጋግጣል

Appendix 4 : python code for WSD based on Cosine Similarity

```
def cosine(phrase, amb):
  folders=[f for f in listdir ("corpus processed" ) if isdir(join("corpus processed", f))]
 N=len(folders)
 for i in folders:
   if trans(i).__eq__(amb):
     folder=i
 files=[f for f in listdir ("corpus processed\\"+folder ) if isfile(join("corpus processed\\"+folder, f))]
 vector_similarity={}
 dot_product={}
 tf_query={}
 for word in phrase.split():
  if word not in tf_query.keys():
     tf_query.__setitem__(word,1)
  else:
       c=tf_query.get(k)+1
       tf_query.__setitem__(k,c)
 df={}
 tf_doc={}
 for i in files:
       tf={}
       dot=0
       corpus=codecs.open("corpus processed\\"+folder+"\\"+i,"r",encoding="utf-8")
       read_corpus=corpus.readlines()
       for j in read_corpus:
```

```
            for k in j.split():
                if k not in tf.keys():
tf.__setitem__(k,1)
                else:
                    c=tf.get(k)+1
tf.__setitem__(k,c)
        for s in tf.keys():
            if s not in df.keys():
                df.__setitem__(s,1)
            else:
                d=df.get(s)+1
                df.__setitem__(s,d)
        tf_doc.__setitem__(i, tf)
  for d in df.keys():
idf=log(N/df.get(d))
    df.__setitem__(d, idf)
vector_query=0
  vector_doc={}
  for j in tf_doc.keys():
    vector=0
    dot=0
    for i in tf_doc.get(j).keys():
        tfidf_doc=tf_doc.get(j).get(i)*df.get(i)
        vector+=pow(tfidf_doc,2)
        if i in tf_query.keys() and i!=amb:
```

```
            tfidf_query=tf_query.get(i)*df.get(i)
            dot+=tfidf_query*tfidf_doc
    dot_product.__setitem__(j, dot)
vector_doc.__setitem__(j,sqrt(vector))
for key in tf_query.keys():
    if key in df.keys():
        vector_query+=pow(tf_query.get(key)*df.get(key),2)
  for doc in vector_doc.keys():
    similarity=0
    for key in dot_product.keys():
        if doc.__eq__(key):
            similarity=dot_product.get(key)/float(sqrt(vector_query)*vector_doc.get(doc))
        vector_similarity.__setitem__(doc, similarity)
  value=vector_similarity.values()
  value.sort()
  value.reverse()
  sense=""
  for j in vector_similarity.keys():
    if vector_similarity.get(j)==value[0]:
        for i in j.split(".")[0].split():
            sense+=trans(i)
            index=j.split(".")[0].split().index(i)
            if index<len(j.split(".")[0].split())-1:
                sense+=" "
  return sense
```

Appendix 5 : python code for WSD based on Jaccard combined with Lesk

```
def lesk_alg(phrase, amb):
  dic_syn={}
  dic_glos={}
  weight=0
  weight_jacard=jacard(phrase, amb)
  for i in phrase.split():
   senses={}
   for j in line:
     token=j.split("@")
     if i.__eq__(token[0]) and token[1]!=None:
      id=1
      for syn in token[1].split(";"):
         if syn[ :len(syn)-2]!="empty":
          gloss_syn=[]
          gloss_syn.append(syn.split(":")[0])
          gloss_syn.append(syn.split(":")[1])
          senses.__setitem__(id, gloss_syn)
          id+=1
      dic_glos.__setitem__(i,senses)
  syn_amb=dic_glos.get(amb)
  weight_gloss={}
  weight_syn={}
  for words in dic_glos.keys():
    for sense_amb in dic_glos.get(amb).keys():
```

```
        count=0
        for sense in dic_glos.get(words).keys():
          for word in dic_glos.get(words).get(sense)[1].split():
               if  word in dic_glos.get(amb).get(sense_amb)[1].split() and words!=amb:
                  if word not in stop:
                      count+=1
          if weight_gloss.__len__()!=0 and dic_glos.get(amb).get(sense_amb)[0] in weight_gloss.keys():
               if weight_gloss.get(dic_glos.get(amb).get(sense_amb)[0])<count:
                 weight_gloss.__setitem__(dic_glos.get(amb).get(sense_amb)[0],count)
          else:
              weight_gloss.__setitem__(dic_glos.get(amb).get(sense_amb)[0],count)
sense_weight={}
for i in weight_jacard.keys():
  if weight_gloss.__len__()!=0:
     if weight_gloss.get(trans(I,split(".")[0])) != 0:
                sense_weight.__setitem__(trans(i.split(".")[0]), weight_jacard.get(i) + log10
          weight_gloss.get(trans(I,split(".")[0])))
    else:
                  sense_weight.__setitem__(trans(i.split(".")[0]), weight_jacard.get(i) +(-1))
value=sense_weight.values()
value.sort()
value.reverse()
for i in sense_weight.keys():
  if sense_weight.get(i)==value[0]:
    sense = i
return sense
```

DECLARATION

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

_____

Dureti Siraj

February, 2017

Thesis has been submitted for examination with our approval as University advisor.

_____

Million Meshesha (PhD)

February, 2017