ADDIS ABABA UNIVERSITY

COLLEGE OF NATURAL SCIENCES

Amharic WordNet construction Using Word Embedding

Mulat Getaneh Tiruneh

A Thesis Submitted to the Department of Computer Science

In Partial Fulfillment of the Degree of Master Science in Computer Science

(Data and Web Engineering)

Addis Ababa, Ethiopia

May 29, 2020

Addis Ababa University

College of Natural Science

Mulat Getaneh Tiruneh

Advisor: **Yaregal Assabie (PhD)**

This is to certify that the thesis prepared by **Mulat Getaneh Tiruneh** titled: *Automatic Amharic WordNet Construction Using Word Embedding* and submitted in partial fulfillment of the requirements for the Degree of Master of Science in Computer Science (in Data and Web Engineering) complies with the regulations of the university and meets the accepted standards with respect to originality and quality.

Signed by the Examining Committee:

Name Signature Date

Advisor: Yaregal Assabie (PhD)_____

Examiner:_____

Examiner: _____

Dedication

For: Brigadier General Asaminew Tsigie

Abstract

A big amount of data is produced on the web and this data is available in the online data portals. By any means, people always need to access, analyze, and organize those data easily. To access and analyze those data effectively there must be an automated system that can understand human language as it is spoken. This is possible by using natural language processing applications. However, most of the natural language applications such as sentiment analysis, information retrieval, question answering, word sense disambiguation, etc. use WordNet as a resource. Some natural language applications like information retrieval can be done using electronic thesaurus and dictionary, but the coverage of such resources is limited. WordNet solves such a problem and it is used as a resource for many other natural language processing applications. A WordNet resource can be constructed using manual, semi-automated, and fully automated methods from the text data. However, while the manual method is time consuming and semi-automated methods are not effective methods since the resource includes different relations in addition with a large dataset. So, using these methods is tiresome and time-consuming. Semi-automated and automated methods can be effective for languages which have sufficient resources like thesaurus, bilingual dictionary, monolingual text corpus, effective machine translator, etc. So, automatically constructing a WordNet resource from unlabeled text data is the best way for languages like Amharic which have limited resource.

In this study, we propose Automatic Amharic WordNet construction using word embedding. The proposed model includes different tasks. The first task is text pre-processing which consists of commonly used text pre-processing tasks in many natural language processing applications. We perform text pre-processing in Amharic text document and train the document using a word embedding gensim library (word2vec) in order to generate word embedding model. The embedding result provides a contextually similar word for every words in the training set. Most of contextual similar words belong to a relation r. The trained word vector model captures different patterns. After training the data we take the trained model as input and discover different patterns that used to extract WordNet relations like: hypernym/hyponym, synonym, and antonym. Conceptual synonym of a word is extracted based on cosine similarity. We use an additional distance supervision method for near-synonym (like meaning exist in dictionary) relation extraction. So, for this method we perform feature extraction task based on given sample seed

words (synonym pairs). In the other hand, we extracted hypernym/hyponym relation from the trained model by taking the advantage of mutual information concept and measuring similarity (based on cosine distance). Whereas antonym relation of words are extracted from the trained word2vec model based on the concept of word analogy.

The common evaluation metrics such as recall and precision were used to measure our proposed model performance. Amharic WordNet prototype is developed and used to tests the system performance of using the collected Amharic text document. Finally, this study shows a result of 78.3% recall and a precision of 53.9. We also evaluate using Spearman's correlation, and achieve +0.79 correlation coefficient.

Keywords: Amharic WordNet, Word Embedding, Distance Supervision, Natural Language processing, Word Analogy, Hypernym, Hyponym, Synonym, Antonym, word2vec, Gensim.

Acknowledgments

Before going into thick of things, I would like to add a few heartfelt words for the people who were part of this thesis in numerous ways. My teachers, and students made valuable suggestions which have been incorporated in this work. It is not possible to acknowledge all of them individually. I take this opportunity to express my profound gratitude to them.

I would like to thank God and gratefully acknowledge the help, guidance and support of Him in my whole life for giving me the wisdom, strength, support and knowledge in exploring things, Secondly, I would like to express my gratitude and I am deeply thankful to my adviser, **Dr. Yaregal Assabie.** (PhD) for his kind cooperation, excellent suggestions, moral support and guidance, continuous help, and assistance during this thesis work. I would like to express my special thanks to my best friends and staff members Yihenew, Ayenew, Tilksew Shiferaw, Tilksew Shmels, Bezawit and Addis Ababa university computer science graduate program teachers for their valuable support throughout the course, advice and follow-ups. And special thanks to my former teacher and friend Fantahun Bogale.

The love, affection and encouragement that I got from my family members is unforgettable, they are my strength, my support and my pride. How can I ever thank my dearest family (***GetFam***) members for always cheering up and spicing up our life. I would like to say thank you to language experts for giving their time to fill my questionnaire form. Finally, I really want to give great respect and credit to everyone who have been involved in my thesis work.

Table of Contents

List of Figures

List of Tables

List of Algorithm

List of Equation

Acronyms and Abbreviations

API	Application Program Interface
CBOW	Continuous Bag of word
KB	Knowledge Base
MT	Machine Translation
MWT	Multi-Word Terms
NLP	Natural language processing
NP	Noun phrase
P	Precision
PWN	Princeton WordNet
R	Recall
SG	Skip-gram
sense2vec	Sense to vector
Word2vec	Word to vector
WN	WordNet

Chapter 1 Introduction

1.1 Introduction

Natural language processing (NLP) is a branch of artificial intelligence that helps computers communicate (understand, interpret and manipulate) with humans in human own language and scale other language related tasks. There are many different types of NLP applications which include machine translation, word sense disambiguation, information retrieval, document categorization, and so on. Most of these applications use WordNet as an important resource [1].

WordNet is a lexical database for natural languages. It groups words of a given language into different sets called synsets, which hold synonym as element. It provides little and brief definitions with examples; it also registers different relations (different implications of the word) among synsets. WordNet for English was introduced in 1985 at Princeton University in the Cognitive Science Laboratory under the direction of psychology professor George Armitage and Fellbaum [2].

WordNet resource is the backbone of many research works of natural language processing applications [3]. WordNets are widely used in various applications of NLP such as information retrieval, machine translation, multi-language information retrieval, automatic text summarization, word-sense disambiguation, automatic text classification [2], etc. It improves the accuracy of many NLP applications.

WordNet construction can be achieved using manual [2], automated [4], and semi-automated [5] methods. Manual construction of WordNet is done through the collection of words with their meaning in a different context (hyponym, hypernym, and synonym) manually. Most automated and semi-automated methods use machine translation (MT) and Princeton WordNet (PWN) to construct synsets of a word automatically and use word embedding approach to extract the context of words within its sense in addition to MT and English WordNet.

Nowadays a new method that can extract synonym of a word is introduced using neural word embedding. Neural word embedding includes word2vec, sentence2vec, and doc2vec. From the list, word2vec is used to generate synsets of a word. Word2vec is a two-layer neural network which is used to process any text document. For a given input text corpus, it outputs a set of vectors. One

of its models is the so-called skip-gram model that represents any words in a text in a real number called a vector. The vector consists of features of a word in N dimensions. Those features are contextually similar words. Word2vec is also used to produce word embeddings. Fatemeh *et.al* [6] proposed a model that reconstructs the linguistic contexts of words using two-layer neural networks.

1.2 Motivation

Amharic is a Semitic language. It ranks as the second to Arabic based on the number of speakers in the world. The federal government of Ethiopia used Amharic language as a working language. In addition to that, it is also the working language of several regional states and cities within the federal government such as Amhara regional state, Addis Ababa city administration, Southern regional state, Dirie-dawa cisy administration, Gambela regional state, and Benshangule regional state. Beside this, Amharic language is being used in governmental administration, public media, and national commerce of the above listed regional states of the country [10]. The majority of Amharic speakers are found inside the country of Ethiopia. But there are also speakers in a number of other countries, particularly German, United Arab Emirates, Italy, Canada, the USA, and Sweden [11].

As a result, many Amharic documents are being produced and stored. To access and effectively process such documents, we need Amharic NLP applications. Amharic WordNet is an important component to enhance the performance of Amharic NLP applications. This has motivated us to develop Amharic WordNet.

1.3 Statement of the Problem

WordNet is constructed for various languages such as English [2], Korean [7], French and Russian [4], Persian [8], etc. WordNet construction can be done in a manual, semi-automated or fully-automated way. Manual construction has some difficulties as it is tiresome, time taking, and requires a large number of lexicographers to hand-build the lexical database [5]. Semi-automated [9] approach uses manually developed simple lexical databases and then advances by extracting conceptual knowledge from resources such as English WordNet, bilingual dictionaries, monolingual text corpora, etc. A fully automated approach uses machine translation and English WordNet to construct WordNet for other languages besides the English language. Machine

translation generates the candidate synsets from English WordNet and uses word embedding method for matching words to synsets. However, due to differences in cultures of societies and linguistic structures, it may not be possible to generate accurate word-to-word translations using machine translation. For instance, one Amharic word may be expressed in three or more English words as in the case of "teseberech"("ተሰበረች") which means "she is broken". Due to complex inflectional morphology, culture-specific meanings, and usages of words and phrases, construction of Amharic WordNet is a non-trivial task.

To our best knowledge, apart from a few sample resources [12], there is no extensive study carried out to construct Amharic WordNet. In this thesis work, we proposed to construct Amharic WordNet from unlabeled text data using the neural word embedding model. We hypothesize that word embedding can automatically extract synsets of a word using their distributional frequency of word that exist in a sentence especially cultural and language-specific words. Furthermore, word embedding helps to extract the contextual meaning of words within its sense.

1.4 Objective

General Objective

The main goal of this study is to construct Amharic WordNet using word embedding techniques.

Specific Objective

- ✓ Conduct literature review on WordNet construction methods
- ✓ Collect Amharic corpus.
- ✓ Identify Amharic word form for WordNet entry.
- ✓ Identify different contexts of a word using word embedding
- ✓ Extract Amharic word synsets
- ✓ Develop a prototype
- ✓ Evaluate the performance of the WordNet.

1.5 Methods

In order to achieve the objectives of this research the following methodologies will be used.

Literature review

We will make thorough review of different related works on WordNet construction. This helps to understand the morphological properties of Amharic, WordNet construction techniques in general and word embedding in particular.

Data collection

The data needed for this research will be collected from the different online and offline data sources. We will use the corpus for extraction of word contexts in different domains and to extract the synsets.

Development of a Prototype

We will develop a prototype of the system used to construct Amharic WordNet

Experimental Evaluation

The accuracy of the constructed Amharic WordNet will be evaluated using different measuring matrices like precision and recall.

1.6 Scope and Limitation

The scope of this work is constructing Amharic WordNet. Among various relations between words, we will consider relations like synonym, antonym, hypernym, and hyponym. The limitation of this thesis work is, it doesn't include the other relations exist in WordNet like meronym, holonym, troponym etc.

1.7 Application of Results

The final output of this research brings great opportunity to enhance most challenging Amharic NLP applications listed below.

Amharic Search Engine

To search Amharic documents, we need a search engine that can handle Amharic queries, written in Ethiopic script. Using Amharic WordNet, we can simply expand the given queries that will increase the performance and accuracy of the search engine.

Word Sense Disambiguation

In natural language, automatic resolution of ambiguity is needed. So, to do this we can use WordNet as resource. Because, WordNet can easily disambiguate the meaning of words since it contains different sense of the word.

Amharic Text Clustering and categorization

Automatic text categorization needs calculating similarities between documents and categories using the information extracted from the document. So, using WordNet we can get semantic relationships between words.

Amharic Information Retrieval

WordNet has to be used as a linguistic knowledge tool to represent and interpret the meaning of word. Text search based on lexical matching of keywords is not satisfactory due to polysemous and synonymous words. Semantic search that exploits word meanings, in general, improves search performance. So, WordNet is important resource of this application since it provides the sematic meaning of words.

1.8 Organization of this Thesis

The rest of this thesis is arranged as follows. Chapter Two provides the detail of different approaches to extract synonym, antonym, hyponym and hypernym. In Chapter Three, we discuss the existing mechanism of WordNet construction for different language. Chapter Four details the system design and the proposed system architecture. In Chapter Five we discuss the system prototype and evaluation metrics and Conclusion of the study presented in chapter six. Possible future extension and contribution of the work also will be presented in Chapter Six.

Chapter 2 Literature review

2.1 WordNet

WordNet (English version) was developed at Princeton University by a group of psychologists and linguists in 1985 for the first time. The psychologists had determined properties that can be transferred from the so-called mental lexicon to lexicography [12]. Unlike dictionaries, what they had in mind with WordNet was searching words conceptually rather than alphabetically. WordNets are taxonomies in which more general top concepts are higher up and more specific sub-concepts under these general concepts are within a hierarchical structure. Every word is interlinked with different relations. In WordNet, words that represent the same concept are grouped together. WordNet can thus be seen as a combination of dictionary and thesaurus [5].

WordNet is the backbone of many research works of natural language processing (NLP) applications [4]. In this section we will discuss the detail of WordNet structure and components and word embedding with its different methods. We also discuss the Amharic language word forms and clustering approaches.

2.2 WordNet and its components

As described in the introduction part, WordNet is a lexical database for natural language processing applications. It groups words of a given language into sets of synonym called synset. WordNet not only groups words into synsets, but it also stores different relationships (word contexts) among words. The main relation types commonly used for natural language processing applications are synonym, hypernym, hyponym, and antonym.

2.2.1 Synonym Extraction

The word that is nearly the same or exactly similar to another word in a language is called synonym. For example, the words talk, converse, speak, and gossip are all synonym of one another [13]. Synonymy is described as symmetrical hyponym [14]. A synonym can be classified into lexical and propositional synonymy, or into lexical, phrasal, and propositional synonymy. The focus of this thesis is lexical synonymy and words that belong in the same class. Synonymy has a different scale of similarity it might be near-synonymy or absolute synonymy or it might be cognitive synonymy as in [15].

According to the Cruse [15] description, if a word is completely similar in all aspects of meanings of words in every context then it is called an absolute synonymy. In cognitive synonym, most semanticists would regard as synonymy. Near-synonym are words whose meaning is relatively less or more similar which is associated with overlapping the sense and meaning. These types of synonym are mostly found in dictionaries of synonym or thesauri.

There are many different models that are used to extract synonym from diverse resources. For instance, synonym are extracted using the following approaches. (1) The automatic extraction method of synonym was begun early to evaluate machine translation (MT) applications automatically. Synonym is extracted based on exact string matches between hypotheses and references [16]. It compares MT output with an expert reference translation in terms of statistics of a short sequence of the word. (2) Through time, the evaluation methods shifted into machine learning approaches [17] to determine the quality of machine translation. Later, some other better approaches such as Meteor [18] (learn automatically for new other languages using bilingual text corpus or dictionary for the development of machine translation system) incorporated the syntactic, semantic, and lexical information and attempted to detect paraphrases and synonym. A synonym may be identified by all possible matches between the sentences according to the following matchers: Exact: // if the surface forms of the word are identical, it is said to be exact match. Stem: // match if the stems are the same as the stem of the given word. Synonym: // Match words if they share association in any synonym set according to the English WordNet database [19].

There are some other approaches to extract synonym using different resources such as a monolingual dictionary, bilingual corpus, and a large corpus of monolingual documents [20]. In a monolingual dictionary, each entry is defined by other words and may also be used in the definitions for other words. This kind of dictionary contains the definition or meaning of words of a single language. For a word in the dictionary, the words used to define it are called hubs and the words whose definitions include this word are called authorities. They use the hubs and authorities of a word to represent its meaning. To say words are similar they must have a common hub and authorities. The authors also use a resource like a bilingual dictionary in order to extract synonym of words. It uses the translations of a word to express its meaning. The assumption in this method is that two words are synonymous if their translations are similar. It is known that the English WordNet is constructed manually. So, it is possible to extract WordNet for other languages using a bilingual dictionary.

The context method is also used for synonym extraction from a large monolingual text corpus of a document [20]. This method relies on the theory that words have similar (synonymous) contexts tend together. They use words that have dependency relationships with the investigated word as contexts. The contexts are achieved by expanding the documents in a tree-like structure called a parsing tree. The result of the parsing is represented by triplet dependency as follows: // <w1, the relation type, w2> where w1 and w2 are the word. For example, a given sentence "I go to school" is represented in triples like this: Word w1 represent as Name <Relation Type, w2> as an attribute, the verb "go" in the above sentence has two attributes <OBJ, school>, and <SUBJ, I>. Thus, the contexts of a word can be expressed using its attributes. In this case, if two words have similar attributes they are synonymous.

Hazem *et.al* [21] introduced a new word-embedding-based approach for the automatic acquisition of synonym of multi-word terms (MWTs) that manage length variability. Their first proposition is an extension of the semi-compositional approach using word embedding to extract synonym of parts of MWT. The semi-compositional word embedding approach is also based on the composition of the elements of MWTs. It can be considered as a variant of the semi-compositional approach introduced (instead of using a dictionary that will provide synonym of each lexical element of the MWT, the method exploits distributional relationships). The difference is in semi-compositional approach, they used distributional approaches were as [21] they explore the two well-known word embedding representations: the skip-gram model and the continuous bag-of-word model (CBOW). Based on the additive property of the word embedding, a fully-compositional approach is proposed as the second is to extract synonym of the entire multi-word terms. The full-compositional word embedding approach aims at extracting MWTs synonym of any length. It provides a joint (additive) representation for all the MWTs which facilitate MWTs comparison. Each MWT is first characterized by an element-wise sum of its word embedding elements. Then, the cosine similarity measure is applied to extract MWTs synonym.

2.2.2 Hypernym and hyponym Extraction

The relationship between hyponym and hypernym is defined as a hypernym is a generic term whereas a specific instance of hypernym is called hyponym. Since hypernym and hyponym always represent a class and subclass respectively, a hyponym is a phrase or word, and its semantic field is more specific than the hypernym (its hypernym). The semantic hierarchy of a hypernym relation

is always wider than a hyponym relation; this is also called as a hypernym is superordinate. Hypernym relation is considered as that consists of the relation hyponym. For example, words such as አየ "Aye", አስተዋለ "Astewale" and አፈጠጠ "Afetete" can also be considered hyponym of the verb ተመለከተ "Temelekete", which is their hypernym. Hyponym and hypernym relations are asymmetric relationships. For instance, Stede *et.al* [22] to determine whether or not it makes sense they test the hyponym/hypernym relation by substituting A and B in the sentence 'A is a kind of B'. For example, 'A banana is a kind of fruit' makes sense but not 'A fruit is a kind of banana'. Hyponym is an asymmetrical relationship. While "each A is a/an B" condition is true, but not the reverse is true. Hyponym satisfies a transitive semantic relation.

Approaches of Hypernym extraction

The different structures of a language can indicate the meanings of lexical items. But the difficulty is finding the relation between words. Because, free text has different forms and contents when compared with the dictionary (which is somewhat regular structure) [23]. However, there is a set of lexico-syntactic patterns that indicate the relation of words. Let us see some patterns with examples:-

NP0 such as NP 1 {, NP 2 .., (and | or) NP i} $i \geq 1$ where NP is a noun phrase.

"A substance agar is prepared by mixing red algae, such as Gelidium, for industrial use or laboratory". From this sentence, we can extract hyponym and hypernym relationships. Since the pattern is NP0 such as NP1, hyponym ("red algae", "Gelidium") and its hypernym is vice-versa.

such NP as {NP ,}* {(or | and)} NP

..Works by such authors as Abel, Adam, and Hwan.

Hyponym ("author", "Abel"),

Hyponym ("author", "Adam"),

Hyponym ("author", "Hwan") the activity of X is kind of Y.

We can also say Hypernym ("Abel", "author").

But it is not always true to assume that the patterns are always accurate to indicate the correct relationship of the word, i.e. noun directly preceding "such as" is the full hypernym if it is preceded by a preposition. For instance, a bearing is a structure that supports a rotating part of a machine, such as a shaft, axle, spindle, or wheel. Previously they also use some other extraction methods, in [24]. Lexico-syntactic patterns are used to extract lexical relations between words from unrestricted text. For example, the pattern may be NP, especially {NP,}* {or| and} NP (where NP is a noun phrase). The approach follows some steps to extract the patterns as follows:-Select manually a representative conceptual relation (hypernym), next collect a list of pairs of terms linked by the previous relation. This list of pairs of terms can be extracted from a thesaurus, a knowledge base, or manually specified. Then find sentences in which conceptually related lemmatized terms occur. These sentences are lemmatized, and noun phrases are identified. They are represented as lexico-syntactic expressions. Finally, find a common environment that generalizes the lexico-syntactic expressions extracted at the above step.

The extraction of hypernym using manually annotated patterns has its limitation. The limitation is, it needs severely identification processes of patterns (which is tiresome), the number of patterns is limited and the number of extracted items are low in number. Due to this reason, to improve the problem encounter, pattern-based method, statistical models, scientific study of algorithms are used to extract hypernym/hyponym relation. Those models are used that perform a specific task in the systems of a computer effectively without using any patterns, inference, and instructions [25]). The machine learning model is automatically swapping the old handcraft knowledge base extraction of relations. This is done via feeding known hypernym pairs first as an example and it identifies automatically huge numbers of lexico-syntactic patterns that are useful. It also achieves a high precision hypernym relation classifier by combining those outlines using a supervised machine learning algorithm [26]. Lin *et.al* [27] represent space for expressing the patterns in order to automatically identify lexico-syntactic patterns. Besides, for the formalization of the space of lexico-syntactic patterns, it uses dependency paths. For semantic processing, Lin *et.al* [27] assures dependency paths are successful to represent lexico-syntactic relations. To represent the syntactic relations among the words, a dependency parser is used. This gives a dependency tree with a list of edge tuples of the form. But to apply the method and to get a better result, highly accurate dependency parser is required.

Besides, the previous approaches also have some other problems. Polysemy issue is avoided by assuming that all words have single `sense altogether and inferring taxonomies explicitly over words and not senses [28]. Enforcing a polysemy to false monosomy leads into erroneous inferences; for example, collapsing the polysemous term bank into a single sense might lead one to infer by transitivity that a blood bank is a kind of financial institution. WordNet synsets are the objects of taxonomy in the case of word relation acquisition. Snow *et.al* [28] proposed a new approach that provides a solution for the above problem.

This problem is fixed by jointly considering evidence about multiple relationships as well as lexical ambiguity within a single probabilistic framework. An efficient randomized algorithm is derived for computing clusters of similar nouns. They learn coordinate terms based on (m; n)-cousin classification probabilistic architecture model. Heterogeneous evidence is integrated by the model from different classifiers. Also, the model has the ability to choose the right word sense and attach a new hypernym. Predicting the relationship of hypernym is acquired by parsing an encyclopedia structured text and newswire text corpus. Using the dependency trees result, for each word pair (a; b), the evidence $E_a{}^H{}_b$ is constructed; from the labeled syntactic dependency path the evidence takes the form of a vector of counts of occurrences and connecting the shortest path found as a and b in dependency tree.

Relation among words like hypernym, hyponym, and meronym can be also constructed automatically using a co-occurrence based approach and pattern-based approach [29]. The pattern-based approach is again classified into the lexico-syntactic pattern method and the dependency pattern method. The pattern-based approach is used to extract hyponym relations from a row corpus. Gathering of the list of terms for each particular relation R is held in order to discover a new relations pattern. The commonalities among the environment and occurrence patterns are searched from the text corpus. First, build a noun pair to identify whether the semantic relation between the pair of nouns. To extract noun pairs from each of the sentences the corpus is preprocessed. The preprocessed step includes sentence segmentation, POS tagging, tokenization, stemming, and noun extraction and pairing. The sentence may contain more than one noun pairs. Only some of them contain hypernym and meronym relationship. Hypernym relationship also is extracted from text corpus proper noun tag using machine learning techniques. This work is done based on dependency parsing with Bayes classifier and ADTree [29].

As we have seen in the above pattern-based methods have been one of the most usable approaches in the extraction of hypernym relation. However, a well-known problem here is, patterns are extremely scattered. If words don't co-occur in exactly the right configuration, no relation can be detected. To improve such type of problem the detection mechanism is shifted to distributional representations. In distributional representations, the representation of a word is in a vector space based on their spreading across text corpus. This method offers rich representations of lexical meaning, improving the sparsity problem, and using the distributional inclusion hypothesis measures similarity to distinguish different lexical relationships. On the other hand, a different automatic approach for the extraction of hypernym relation using dictionary terminology is also proposed by Baisa *et.al* [31]. It is a specialized corpus-driven method and a term similarity-based approach. The specialized corpus-driven approach is based on the pattern whereas the similarity-based approach is finding hypernym of a term by searching the term database. Then the given term is compared to all existing terms in the system database. Then a better hypernym/hyponym candidate is identified by their most similarity.

Recently, the word embedding model is the main focus area of research, especially for lexical relations extraction. Word embedding is a predict-based neural network model that predicts the context words or a "count-based" traditional distributional similarity method combined with dimensionality reduction on and occurrence of the word in a text corpus. The method merges dimensionality reduction with the old distributional similarity method [19]. Fu *et.al* [30] also propose a novel approach for semantic hierarchy construction using word embedding. The word embedding represent the terms of text corpus in a real-valued low-dimensional vector. It is also identified as a distributed representation of the word. Word embedding preserves linguistic regularities of terms such as the contextual meaning of words. Word embeddings have been empirically shown to preserve linguistic regularities, such as the semantic relationship between words. The aim is to identify hypernym-hyponym relations using word embeddings, which have been shown to preserve good properties for capturing the semantic relationships between words. Using word embeddings, they learn the hypernym-hyponym relationship by estimating projection matrices that map words to their Hypernym. Further improvements are made using a cluster-based approach to model more fine-grained relations. Then a few simple criteria is used to identify whether or not the new word pair is a hypernym-hyponym relation. Based on the pairwise hypernym-hyponym relations, they build semantic hierarchies automatically. In this paper, the

researcher used the skip-gram model. Skip-gram is a model for learning word embedding that efficiently induce word embedding. They conclude that for the identification of the semantic relationship between a word, skip-gram model achieves the best accuracy than others.

2.2.3 Antonym Extraction

Antonym means words that have the opposite meaning. For instance, the antonym of the word ‘hot’ is ‘cold’. The word antonym is derived from two words ‘anti’, which means ‘against’ and the word ‘onym’, which means ‘name’ as defined by Thalenberg *et.al* [32]. This type of relationship is an asymmetrical relationship. Terms can be replaced with each other in the pattern like "neither tall nor short" or "neither short nor tall".

The current research trend is based on the premise that words with similar meanings occur in similar contexts. Computational resources have been used to improve and implement vector space models. Thalenberg et.al [33] proposed the establishment of two types of similarities. These are first-degree co-occurrence, or sintagmatic association, between words that frequently appear near each other and occur in similar contexts. The contextual similarity between two given words can be calculated by the nearness of their vectors after created. The problem with the notion of semantic proximity is: it is impossible to distinguish between synonym and antonym of a word. The solution is, most of them need external resources, such as an electronic dictionary or fixed sematic patterns such as x to, y or x and not y, etc. But if the purpose is to create thesauri and WordNets automatically, they do not need one to work properly. So, to solve such a problem, they apply another method. "All parts of speech are useful for distinguishing antonym from synonym [33]," verbs being the most useful one. Applying skip-gram, they can find word embedding and context embedding encoded into vectors. The vector consists of a high dot product for the neighboring word and vice versa.

For antonym and synonym pair, Scheible *et.al* [34] filter the most frequent adjectives in the corpus and list antonym and synonym manually using paper-based thesauri. Finally, the researchers will verify if these methods are enough to determine the distance between antonyms using a classifier, based on a minimum score for the cosine similarity, obtained through the experiment. A distributional similarity model is not only used to retrieve synonym words but also antonym words because of the distributional similarity of synonym and antonym. Scheible *et.al* [34] proposed a

model of word-space that distinguishes synonym and antonym relations. The word space model exploits window-based features for synonymous and antonymous adjective pairs. The assumption about antonymy is that even though antonymous words are said to be opposites, they are nevertheless semantically very similar because they share the same dimension. Antonymy exists in the same sentence, for instance in contrastive constructions such as 'either x or y'. Especially direct antonym (such as cold/hot, strong/weak) exist interchangeable in most contexts. The pairs of a word occurring in contrasting or opposite meaning in the thesaurus of the dictionary are marked as an antonym. Cosine values separates synonym from antonym, with antonym in the lower-value and synonym in the higher-value partition.

2.3 Word Embedding

2.3.1 What is neural word embedding

Neural word embeddings are the texts converted into numbers and there may be different numerical representations of the same text. Word embeddings refers to low dimensional, real-valued dense vectors encoding the semantic information of words. Word embeddings are a vectorial representations of the meaning of words. It also acquires more complex geometric structures as a side effect of some algorithms. An example of this is real-world analogies that can be discovered using simple vector arithmetic's: king - man + woman = queen. Word embedding trained on the most popular algorithms learn these vectorial representations just by scanning big corpora, but it can be trained also on knowledge graphs. Both of these algorithms rely on a single assumption: words that appear in similar contexts have similar contextual meanings. The concepts word embeddings, distributed word representations, and dense word vectors can be used interchangeably. Word2Vec is the first model for learning word embeddings from unlabeled data. Word2vec involves two different models, namely, CBOW and SG:

- CBOW (Continuous Bag-of-Words): Using context words to predict the middle word.
- SG (Skip-Gram): Using the middle word to predict the context words [35].

Word2vec is a neural network algorithm. It captures the semantics of words in a text which you can get out of it [36].

Word embedding captures the meaning of words into a dense vector and it is a better way to encoding the meaning of words because it captures semantic properties as well as the syntactic features about the word in the created dimension. Word embedding can be represented via hybrid representation, which is called SeVeN (Semantic Vector Networks) [37]. Similar to semantic networks, they use a graph-based representation in which nodes are associated with words. In contrast to semantic networks, however, these edges are labeled with a vector, meaning that relation types are modeled in a continuous space.

To attain an appropriate relation vector for the given two words a and b, the vector representations can be found from a pre-trained model of the words that appear in sentences. They use autoencoder to obtain a vector representation which only reflects the words that relate to the relationship. Once the semantic vector network has been learned, they use it for various aspects. For instance, the relation vectors could be used for identifying words that have a specific lexical relationship such as hypernym/hyponym, for measuring the similarity among relational or complementing open information extraction systems. Luis *et.al* [37] evaluate the potential of semantic vector networks based on two tasks: for enriching the word2vec as the input to the architectures of neural network and for semantic similarity modeling using unsupervised learning. They have evaluated their model in terms of their effectiveness in two standard natural language processing applications: text classification and word similarity. In both cases, they achieved above the baselines that use standard word vectors alone.

Generally, word embeddings are classified into two groups: prediction based embeddings and frequency-based embeddings [38].

2.3.2 Frequency based Embedding

In this category there are generally three types of vectors included: count vector, TF-IDF vector and co-occurrence vector.

A. Count Vector

Before modeling each document, it learns the vocabulary from the given documents. The modeling is by finding the frequency of the word in the document. For instance, Given a document D and V the number of unique words in the vocabulary, then the size of the count vector-matrix will be given by D*V. Let us take the following examples: - Doc1: "The book is on the table" Doc2: "The

boy is crying". From these documents the vocabularies are {the, book, is, on, table, boy, crying} and D = 2, V = 7 so the matrix is 2X7 number of document * number of unique words.

	The	Book	Is	On	Table	Boy	Crying
Doc1	2	1	1	1	1	0	0
Dco2	1	0	1	0	0	1	1

B. TF-IDF Vector

This is another method which is based on the frequency method next to the count vector. It takes into account the sense of a text from the occurrence of a word in the entire corpus. Some words may present high frequencies in every document (e.g. "the", "an", "a", "is" in English), those words are low information-bearing (stop-word) in the actual contents of the given document. Terms that are very frequent shadow the frequencies of rarer and more interesting terms. This method removes such a problem by representing or assigning them a lower weight.

C. Co-Occurrence Vector

Words co-occurrence matrix method describes how words occur together. A contextually similar words tend to occur together and this model captures the relationships among those words. By counting the number of word co-occurrences in a given text corpus, it computes words co-occurrence matrix. To produce more accurate word vector representations, first it preserves the semantic relationship between words. The drawback is requiring a big memory size to store the co-occurrence matrix and factorizing matrix is out of the system.

2.3.3 Prediction based Vector

These methods were a prediction based on the sense that they provided probabilities to the words and proved to be state of the art for tasks like word analogies and word similarities. Word2vec model is used today to generate word vectors.

Word to Vector

It is prediction-based machine learning model which produces a low dimensional real valued vector called word embedding. Word2vec is a two-layer neural network that is used to process text. Its input is unlabeled text corpus and its output is a set of vectors which consists of contextual meaning of words. It turns text into a numerical form that deep neural network can understand but the word vector by itself is not a deep net. It factorizes a word-word matrix that contains co-occurrence counts. In word2vec factorization is done by scanning the corpus with a given window (W) size. The center word in the given window is called target (T) word and a few neighboring words in the left and right of the target word are called the context (C) words. Word vector model starts finding unique vocabulary in a text corpus and then the set of word vectors are initialized randomly and perform a linear scan in the whole document. Then, apply dot products between words and their contextual related words. Besides, it tries to minimize this metric performing SGD (stochastic gradient descent). Each stochastic sample is a consecutive window in the corpus. In each step, a target word T is surrounded by context words, their vectors are pushed together slightly, depending on the learning rate. The model is very simple and efficient way to capture contextual meaning of words and it can be applied in a very huge text corpus for any language.
To address this Word2Vec model with hierarchical softmax regulator was proposed for the first time. Later on, they proposed an optional method which is negative sampling. The optional negative sampling method has been shown to be more effective and simpler for small size data. The basic premise is that each time few random words are sampled to minimize their distance between vectors [38].

The word2vec algorithm uses a feed-forward neural network that used (is an artificial neural network where in connections between the nodes do not form a cycle) to predict the vector representation of words within N-dimensional language model. Word2vec model has two types of models: those are continuous bag-of-words (CBOW) and skip-gram (SG). Word2vec semantic representations were generated with the open source Gensim Python library [39]. It has different parameters that can be set based on trainers need. For instance, window size, the number of negative samples, model type, and the representation number of dimensionality are parameters of the model and used to maximize the performance. Most of the parameter usually were set to default gensim values except those listed above. Using the gensim similarity model, the semantic similarity of words is measured. The given vectoral representation between two words is calculated as follow cos (v1, v2) [40]. The semantic distances between two words d (v1, v2) was calculated

as 1 minus the semantic similarity (d (v1, v2) = 1 – Cosine Similarity (v1, v2)) which is also known as cosine distance.

A word vector represents every word with one 1 at the index of the word and all other zeros, an example of vector representation is show below.

$$W^1=\begin{pmatrix}1\\0\\.\\.\\0\end{pmatrix} \qquad W^2=\begin{pmatrix}0\\1\\.\\.\\0\end{pmatrix} \qquad W^n=\begin{pmatrix}0\\0\\.\\.\\1\end{pmatrix}$$

Here, all ones represent the words at index 1, 2… up to n. Then, using word co-occurrence counts find the embedding. The co-occurrence of words can be counted via the window-based matrix. The method counts the number of times each word exists inside in the defined windows size [41].

Continuous Bag of Word Model

The simplest version of the continuous bag-of-word model (CBOW) is introduced in [35]. Unlike skip-gram, the model is using the surrounding words it predicts the target word. A simple CBOW model is shown in Figure 2:1.

In the Figure 2:1 N and V represent the Number of neurons in hidden layer and the vocabulary size respectively. The input is a one-hot encoded vector, means that only one out of vocabulary units will be a value of one from the list of given context words and the rest units are 0. The weight between input and output layer is represented by N x V matrix W. The hidden layer H is the transpose of matrix W, H➔ W^T. Finally, from the hidden layer H, it generates an output layer. It uses softmax, log-linear classification model to acquire the multinomial distribution or the next distribution of words.

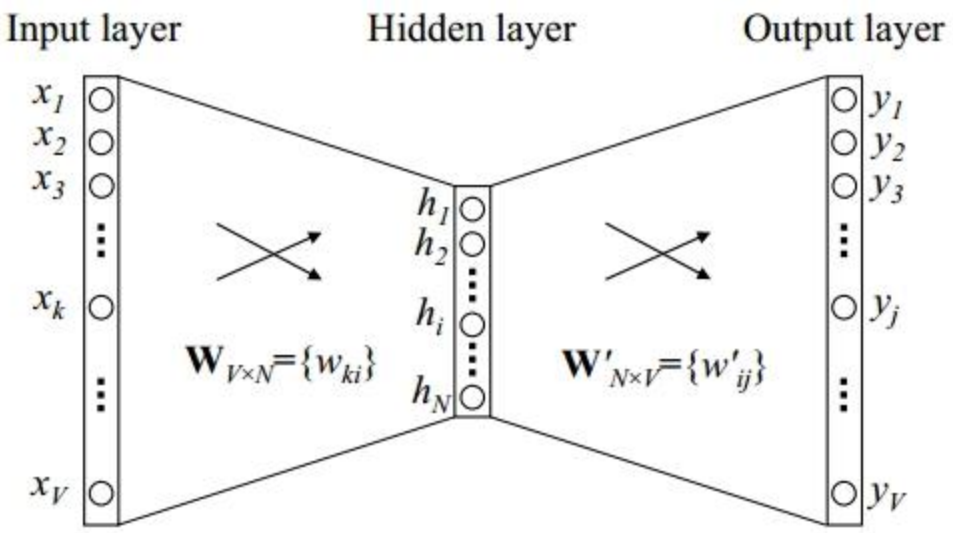

Figure 2:1 A simple CBOW model with only one word in the context

Skip- Gram

The skip-gram is also a type of word vector model introduced by Mikolov [35]. Skip-gram (SG) model is the reverse of CBOW, which means the input layer contains the target word, and the output layer gives the context words for the current target word. The model predicts the contexts based on the central target word. The skip-gram model shows better performance in [35] semantic tasks.

In skip-gram model VwI is used to denote the input vector of the only word on the input layer. The hidden layer is the transpose of the input layer, H = WT (k;*)= VwTI . On the output layer, instead of outputting one multinomial distribution, the output is C multinomial distributions, c = 1; 2; ***; C. figure 2:2 shows a skip-gram model.

Since those models are probabilistic in nature and prediction based methods, they are preferable to perform better to deterministic methods. And these are low on memory consumption. As described in the above, CBOW can use many context words to predict the 1 target word. Essentially continuous bag of word is best when an input text corpus is not so huge. On the other hand, the skip-gram model is more fine-grained than CBOW, and it captures infrequent word

contextual information and able to extract more information. Basically, this embedding's model is more preferable and accurate when it is applied in large corpora.

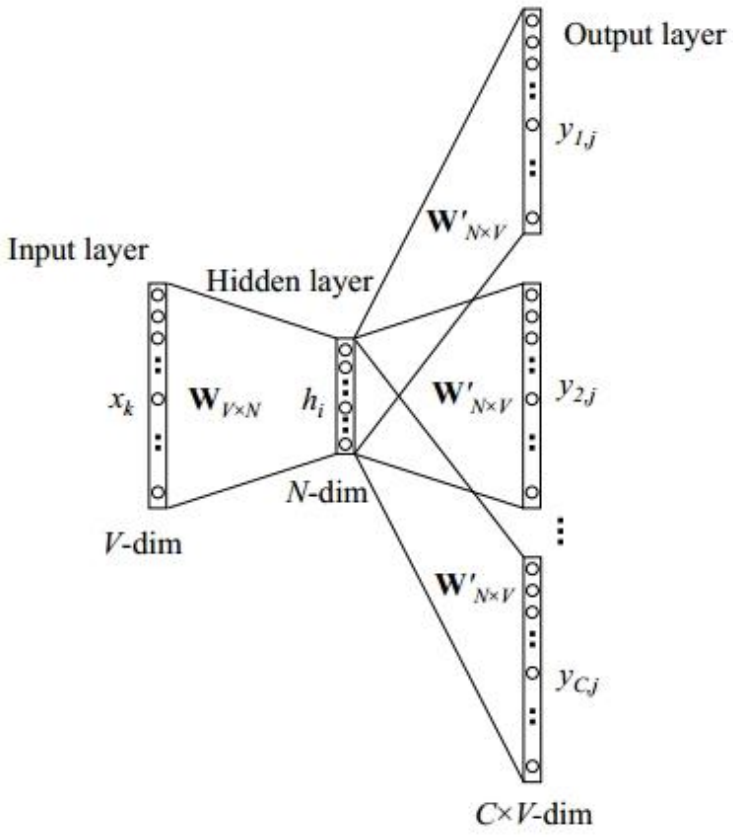

Figure 2:2 A simple skip-gram model.

2.4 Amharic Language

Amharic/አማርኛ is a member of the Semitic language branch of the Afro-Asiatic family. It is spoken in almost all parts of Ethiopia and is the working language of the government of Ethiopia. The name Amharic (አማርኛ) comes from the northern and central Ethiopia, which is thought to be the classical, historic, and ecclesiastical language of country Ethiopia. The language has its alphabet, called ፊደል/ fidäl, inherited from the Geez (Ethiopic currently Ethiopian Orthodox Church) language.

Geez is an ancient South Semitic language which now is used only by Ethiopian Orthodox Tewahedo Church. Written Ge'ez can be traced back to at least the 4th century A.D. In modern

Ethiopic script, each syllograph (syllable pattern) comes in seven different forms (called orders), reflecting the seven vowel sounds [43].

Fidäl is a syllabary writing system where the consonants and vowels co-exist within each graphic symbol. Unlike the majority of its Semitic scripts, fidäl is written from left to right. The writing system consists of 33 consonants, each consonant having seven "orders‟ or shapes depending on the vowel with which a given consonant is combined and some additional orders of fidäl are called ዲቃላ ሆሄያት/dikala hoheyat for some consonants only. It is necessary to have a Unicode encoding before one attempts to make use of the fidäl. Because of the syllabary features, due consideration must also be given while dealing with Amharic strings, since there is no standard method that transliterate Amharic hoheyat into the Latin alphabet [43].

As many other languages, Amharic language words have also their morphological syntaxes. In Amharic nouns decline for number (plural/singular), species (indefinite/definite) and case (genitive/nominative/accusative/dative). Unlike these variable features, the nouns exhibit an inherent behavior towards gender, that is, a given Amharic noun is either feminine or masculine. These declensions are usually achieved through affixation. The most predominant affixations is suffixes while there are also other good affixations like prefixing's. The above order as well shows how these nominal affixes appear. We show this by giving an example 'the books' (plural, definite, noun phrase used as a direct object) from the sentence 'I sold the books.' and show Amharic equivalent for a given phrase as follows, the books - መፅሐፎቹን - metshaf-och-u-n.

Here, the suffixes ('-och', '-u' and '-n') are added to the head noun 'metshaf‟, which means book, to mark declensions for number (Pl) and definiteness (Def) to get the required form – the books - metshafochun ("መፅሐፎቹን"). The pronouns in Amharic can be put into three persons as in English, but there are some unique features such as the second person 'you' which may take different agreements when referring 'singular-male' or 'singular-female', 'plural', 'respected (politeness)', nouns as shown below.

You (Masc. Sg.)	:	አንተ - Ante
You (Fem. Sg.)	:	አንቺ - Anči
You (Pl.)	:	እናንተ- ənantä
You (Politeness)	:	እርስዎ- ərswo

Adjectives come before nouns in a sentence to modify them. In a sentence '*anchi bäTam gobäz tamari näsh*' – 'you are a very clever student' there are two adjectives 'bäTam' - *very* and 'gobäz' – *clever*. 'bäTam'- *very* modifies 'gobäz' – *clever*, and 'gobäz' modifies the noun 'tamari' – student. Amharic words are constructed by linearly concatenating morphemes, Semitic languages have unique non-concatenative properties in addition to the conventional concatenative modifications. Therefore, in Amharic, a verb, which is the most complex category of words, is created generally from consonantal radicals which are inflected by a process of merging with vocalic components based on various patterns. The majority of roots, like other Semitic languages, have three radicals. However, there are also a significant number of verbs that are multi-radical and bi-radical. A verb takes various forms depending on the voice, tense-aspect-mood, and root structure while inflecting for person gender and number. This is done through the addition of affixes at both ends and even between the roots of a stem. The numerals in Amharic can assume a cardinal or ordinal form. The ordinals are all the times achieved by adding the postfix *ăňňa* on the cardinals.

Cardinal Ordinal

አንድ ānidi*/ one* አንድ -ኛኛ -> አንደኛ *ānidi-ăňňa/ first*

The prepositions appear as simple prepositions that are stand alone or as separate entities coming both at pre and post positions.

To – you	ለ - አንተ - *lä* antä	(pre)
On - you	አንተ - ላይ - antä *lay*	(post)
With you	ከ - አንተ - ጋር - *ke* antä *gar*	(both pre and post)

2.5 Clustering

Cluster is a collection of data objects in which unrelated (or dissimilar) to the objects in different groups and similar data objects are grouped in the same cluster.

Cluster analysis is a means of finding similarities across a given data according to the features found in the given data and grouping similar data objects into a single cluster. The grouping of similar elements can be done using different approaches; the major clustering approaches are hierarchical, partitioning, and density-based approaches. Each approach has different techniques and methods to perform clustering [44].

The partitioning approach constructs various partitions and then evaluates them by some criterion. The approach is partitioning a database DB of n number of objects into a set of clusters k. The typical methods of this approach are k-medoids, k-means and CLARANS.

Hierarchical approach creates a hierarchical construction and decomposition of the set of data using some criterion. It uses a Euclidean-distance matrix as a clustering criterion. The method requires only the number termination condition which means the user does not specify a value of k. But, the algorithm constructs a tree-like hierarchy which implicitly contains all values of k. The hierarchy has two ends, leaf at the bottom and roots at the top. On the leaf, there are n clusters each containing a single object, and on the root of the hierarchy, there is only one cluster containing all n objects. Basically, this hierarchical clustering is performed using the agglomerative method (Agnes) and divisive methods. Initially, it considers a single element as a cluster.

A density-based approach is based on connectivity and density functions. Typical methods are: OPTICS, DBSACN, and DenClue.

The need for clustering is for prediction based on groups (patterns recognition), data reduction (association analysis, classification.), hypothesis generation and testing, to finding K-nearest Neighbors, etc.

2.6 Summary

In this chapter, we discussed the WordNet itself and the different components of WordNet. We call the different relationships as components of WordNet. In the chapter, we discussed the different existing extraction mechanisms of synonym, hyponym, antonym, and hypernym. A synonym can be extracted by using machine translation (from English to any other), monolingual dictionary, bilingual corpus etc. Hyponym shows the relationship between a generic term (hypernym) and a specific instance of it called hyponym. The hypernym relation can be extracted: using lexico-syntactic patterns, dependency parser, and using word embeddings.

Word embedding refers to low dimensional, real-valued dense vectors encoding the semantic information of words. Word embedding is done by using word2vec model which involves two different models namely continuous bag-of-word and skip-gram.

In addition, we also discussed the cluster analysis and the different approaches of clustering. The major clustering approaches are partitioning hierarchical and density-based approaches. From

those approaches, a hierarchical approach creates a hierarchical construction and decomposition of the set of data (or objects) using some criterion.

Chapter 3 Related Work

3.1 English WordNet

Fellbaum *et.al* [2] describes the first manual construction of English WordNet. They take the major relation synonymy among words in WordNet. Synonym words which describe similar concepts are arranged into sets called synsets. Using a small number of conceptual relation, its 117,000 synsets are connected to other synsets.

3.2 Chinese-English WordNet

Chen *et.al* [45] proposed the construction of Chinese-English WordNet. They use five linguistic resources ASBC corpus, Cilin, Semcor, Chinese-English dictionary, and WordNet to build a Chinese-English WordNet.

First, in the construction, they put unambiguous words into WordNet by looking up a Chinese-English dictionary. Although these words do not have translation ambiguity, the corresponding English translation may cover irrelevant senses besides the correct one. They find the most similar synset from each English translation.

When they translate a word if the English translation corresponds to only one synset, this synset is the solution. Else, if the English translation resembles to more than one synset, POS information is considered as below:

(a) If the Chinese sense tag belongs to one of the categories (Cilin lists the words based of lexical categories, e.g., A-D (nouns), E (adjectives), F-J (verbs), K (auxiliary), and L (the others)) A-D and there is only one noun synset, then the synset is the solution. Otherwise, they measure the similarity between the Cilin sense vector and each WordNet synset vector using the cosine formula.

(b) If the Chinese sense tag belongs to one of the categories F-J, they try to find a verb synset in a similar way as in (a). If it fails, try noun and adjective synsets instead.

(c) If the Chinese sense tag belongs to category E, try adjective, adverb, noun, and verb synsets in sequence.

(d) If the Chinese sense tag belongs to category K, only adverb synsets are considered.

The above algorithm is to work for unambiguous words only. So, here they consider the ambiguous words.

That is, they have more than one Cilin senses. Chinese-English dictionary lookup finds all the English translations. WordNet search collects the synset candidates for the translations. Some synsets are selected. Here the problems of translation ambiguity and target polysemy must be faced. In other words, not all English translations cover all the Cilin senses. Because the Chinese-English dictionary does not distinguish Chinese senses and English translations. It shows a problem when translation ambiguity and target polysemy are integrated together. Here, they take a conservative way. At first, the synsets, e.g. Synset11, synset12, synset1n.. synset41 synset42.. synset4m, are collected and partitioned by POS. Then, they select a synset from synset candidates for each sense of the Chinese word by using the similar way as mapping unambiguous words. Finally, the result is tested in Chinese-English information retrieval. The CLIR model in monolingual information retrieval with smart information retrieval system, TREC topics 301-350 and TREC-6 text collection achieve 69.23% performance. It also gains 10.02% rise in comparison to a model that resolves together target polysemy and translation ambiguity by a Chinese-English dictionary.

3.3 Construction of Portuguese WordNet

Marcelo *et.al* [46] present the Brazilian Portuguese WordNet process build from the original Princeton WordNet 3.0. Automatically translating the Princeton WordNet using Google's online translation resources is the technique they use to construct Brazilian Portuguese WordNet. In the construction, they expand existing Brazilian Portuguese WordNet rather than starting from scratch. To do this, they use six steps, which are assigned to specific tasks of computing, and the explanation of each step as follows

Preprocessing is the first step in the process used to split the organization of PWN 3.0 in two sets of concepts, which then form two groups of four files implemented in two types of files. Each one of these files is related to its grammar class (adverb, adjective, noun, and verb). The first set refers to indexing terms for each row, there is first element which is a concept, followed by the synsets which belong, parted by spaces. Also, in the second set of files, there is a first component in each record that is the index of the synset, followed by the relationships and concepts. Then using the first set of files from PWN 3.0, the concepts in these files provide a listing of candidate terms for

the translation. For translation, they use Google Translate service com.google.api.translate class. Post-processing is the third step which is used to eliminate noise due to misconceptions in translation. It deletes concepts that require valid translation. To check if the concepts translated, the synsets formed and relations were verified also in another thesaurus we use the validation step. The evaluation step consists of the generation of reports containing information about the translated base itself, such as totals of translated terms, synsets, and relationships. Finally, edition is allowing editing and manipulation of information. The system shows an average of 61% of translated elements.

3.4 Construction of WordNets Synsets for any Language Using Machine Translation

Lam et.al [47] propose an approach to create WordNets synsets for a target languages T using Machine Translation and/or a single bilingual dictionary translating existing WordNet (PWN). To create the synset candidates they use different approaches those are:-the direct translation (DR) approach, intermediate WordNets (IW), intermediate WordNets and a dictionary (IWND). They take advantage of the fact that every synset in PWN has a unique offset-POS (part of speech), referring to the offset for a synset with a particular POS from the beginning of its data file. Each synset may contain one or more words. The main idea is to extract corresponding synsets for each offset-POS from existing WordNets related to PWN and translate to target T using machine translation. Then, they find the correct words for a specific offset-POS in T by applying a ranking method on these candidates.

In the DR approach to extract words they translate WordNet into the target language to create translation candidate synset for each offset-POS. They use the IW approach to handle ambiguity in synset translation. For each offset-POS, they extract the synsets from intermediate WordNets. Then, translate from intermediate languages to T (Target language) using MT to generate a synset candidate. The IWND approach is like the IW approach, the only difference is IWND translate synsets extracted from intermediate WordNets to English, and then translate them to the target language. The experimental evaluation shows the different percent of coverage for different languages. For instance, for language ajz, asm, arb, dis and vie 18.60%, 64.87, 65.95%, 20.51%, and 83.47 % respectively.

3.5 Constructing a Probabilistic Persian WordNet

Fadaee et.al [48] presented a fully-automated approach for constructing a probabilistic Persian WordNet. The main aim of this paper is to construct a scalable Persian WordNet with better quality by defining a Bayesian Inference for estimating the probabilities of links between words and synsets. In this work, they create a probabilistic WordNet. The link between a word and its synsets has a probability assigned to it that signifies the relatedness of each synset to the word. The method consists of the following steps. Collect candidate synsets as possible senses for each Persian word is the first step. Using a bilingual dictionary they find all possible definitions of a Persian word in English. And then find all senses of the English words from Princeton WordNet. And they consider them as potential senses of the Persian word. Next, compute the probability of relatedness of each synset to a particular word. They use word sense disambiguation technique to assign the right sense of each word based on the context in which the word appears. During sense disambiguation of words, they compute the probabilities of assigning variety senses to a word. Finally, they eliminate the senses assigned small probabilities according to some measures and take the highest word-synset probabilities as the final WordNet. Their experiments show that higher precision in comparison with existing automatically-built Persian WordNet

3.6 Constructing Myanmar WordNet

Phyue et.al [49] present a method of constructing a Myanmar WordNet lexical database from English WordNet and Myanmar-English Machine Readable Dictionaries (MRDs). The method has three phases which contain the MRD extraction, the link analyzing, and the WordNet construction phase. In this phase, data is extracted from Myanmar-English MRDs and WordNet. They convert the data (WordNet) from multiple resources with different formats into a single format and according to their POS joins and aligns the scattered data for smoothly access and group the data. They extract data from Myanmar-English MRDs. And they apply format conversion, font conversion, merging MRDs, noise duplicate and missing word removal, and data grouping by POS.

The link analyzing phase analyzes classifies and generates candidates for translation links. The translation link considered to as the relationship between English and Myanmar Word. The candidate link is considered as a relationship between Myanmar's word and their meaning. In the construction phase, they were constructing Myanmar WordNet from the verified translation link

and WordNet. In their, Myanmar WordNet has coverage of 173853 links, 38142 synsets, and 25865 words.

Tarouti et.al [50] propose a method to increase the quality of automatically created WordNets by using word embedding to perform similarity computation. They start by automatically generating WordNet synsets for a target language T using translating synsets from several intermediate WordNets and rank them.

To validate the synsets that are created using translation and in order to obtain relations between them, they use the word2vec algorithm to generate word representations from an existing corpus. They generate representations of words using skip-gram and a continuous bag of word models with several different vector and window sizes to obtain the settings for the highest precision. To remove irrelevant (false word members within synsets) words in synsets they compute the cosine similarity between word vectors within every single synset in TWN. To do this they define a threshold α and cosine similarity. Large α is a candidate synset of TWN. They also compute the cosine similarity between words within pairs of semantically related synsets, to verifying the constructed relations between synsets in TWN or validating candidate relation. The similarity computation process and algorithm are the same as the above. The experiment shows the precision of the synonym, hypernym, part holonym, and member-meronyms they produce is 78.4%, 84.4%, 90.4%, and 79.6% respectively, with the threshold set to 0.288.

Bond et.al [51] Propose extending the method for Japan WordNet. The extending is achieved by increasing the coverage, linking it to examples in corpora, and linking it to other resources (Suggested Upper Merged Ontology and the Japanese semantic lexicon GoiTaikei). Increasing coverage is done in two ways. The first one is correct automatically translated synsets manually or by comparing it with another resource. They add missing words for the poor candidate and add new synsets for Japanese concepts that may not be expressed in the English WordNet (language-specific words). Then they link the Japanese WordNet to the following resources: Suggested Upper Merged Ontology (SUMO); GoiTaikei, a Japanese Lexicon; and a collection of pictures from the Open Clip Art Library. For SUMO they used existing mappings. Whereas for the others they find confident matches automatically and then generalize from them. Finally, the result shows that the WordNet contains an estimated 5% errors and it did not fully model the lexical structure of Japanese words.

3.7 Construction of Persian WordNet using supervised learning method

Zahra *et.al* [52] discuss the construction of Persian WordNet using a supervised learning method. They use Princeton WordNet, bi-lingual dictionary, pre-existing Persian WordNet, FarsNet, and a Persian corpus as resources. The English concepts are represented by one synset in PWN. To expand, it is considered that for the most concepts in English, there exists an equivalent concept in Persian without the language specific concepts. Hence, Persian synset representing a similar concept as the English can be constructed by identifying the correct translations of an English word found in the individual synset. The first step for WordNet construction is lemmatizing words to adapt the different forms of the words and translating the Persian words by Aryanpour (a bi-lingual dictionary) to English counterparts. Then using WordNet identifies English synsets of each Persian word. But all synsets are not important because of the polysemy of English language word (two Persian words may have the same synset). Due to this they remove false links using Bijankhan corpus is enriched by POS tagging of Persian word and from 247,947 links 47,291 are removed and a total of 200,656 candidate links have remained. Their experiments on the Persian language show it outperforms the previously proposed automated method, with the precision value of 91.18% for correctly classified links. The traditional methods (the first method) of manual construction (Used originally in PWN) suffer a series of drawbacks. Due to this problem, WordNet for different languages was constructed by using a translation of English WordNet. Portuguese WordNet offered a method of automatic translation of Princeton WordNet (PWN) version 3.0 [53]. The Portuguese language have the following existing resource PAPEL, MWN.PT, TeP 2.0 /WordNet.BR, Port4NooJ and WordNet.PT. Tep 2.0 was developed in the assumptions of PWN. It include19, 888 synsets and 44,678 lexical units, taking an average of 2.5 units per synset. PAPEL is a resource used like a dictionary. Its contents are semi-automatically found, using MindNet project as inspiration. Commercial and academic license are the two different marketing method of the MWN.PT. Also it is available online from the official site. The version 1 of MWN.PT MultiWordNet of Portuguese include 17,200 synsets validate manually. It similarly include 16 thousand lemmas about Brazil and the Portuguese of Portugal, and more than 21,000 word forms, meanings.

WordNet.PT (WN.PT) is a lexical knowledge database of Portuguese. It is constructed manually under the General model of EWN. The integration of WN.PT on the multilingual network model

of EuroWordNet (EWN) is done using Inter-Lingual-Index (ILI). The ILI is a list of synsets, corresponding to concepts mostly extracted from PWN 1.5 version. Port4NooJ is a set of resources developed in the linguistic environment NooJ for automatic processing of Portuguese. It consists of dictionaries and grammars, which may also be used in applications such as automatic translation from Portuguese into English.

Several works have been done using heuristics and algorithms, In line with these principles, in paper [53] the author's presents a proposal for automatic translation of PWN 3.0 to Portuguese of Brazil. Here they expand rather than starting from scratch. There are many steps in the construction of Portuguese WordNet from PWN.

Shamsfard *et.al* [9] developed a lexical ontology for Persian called FarsNet. A hybrid semi-automatic approach was used to acquire lexical and conceptual knowledge from resources such as semantic templates, bilingual dictionaries, monolingual corpora, WordNet, and morpho-syntactic. The approach is incremental to construct FarsNet, it develop a kernel and it extend in a semi-automatic way. The acquisition approach consists of the following main steps: first providing initial resources, then developing an initial lexicon based on WordNet, and performing WSD. To develop an initial lexicon three separate approaches were used in parallel. Manually gathering a small lexicon about 1500 verbs and 1500 nouns, some heuristics, and word sense disambiguation (WSD) methods have been used to find the most likely related Persian synsets. First, a small kernel containing just the base concepts are manually created and then the initial big lexicon is created. Then, new knowledge from available resources is extracted. After creating the initial lexicon, extra words are gathered from a tagged corpus and assigned to a synset. The manual evaluation of the resulting Persian lexicon shows an accuracy of about 70%.

Sathapornrungkij et.al [5] propose a semi-automatic approach to build Thai WordNet from LEXiTRON machine-readable dictionaries and existing WordNet. Thai WordNet synsets are derived from the PWN. The translation relations between English and Thai words are offered by LEXiTRON whereas WordNet provides semantic and lexical relations between English words. Semantic links that are obtained from WordNet are used to derive the candidate links between Thai synsets and words. LEXiTRON is used to found the translation links. To derive links between Thai words and ENGLISH WORDNET synsets, 13 criteria have been used which are categorized into three groups: monosemic, polysemic, and structural criteria. Monosemic criteria emphasize

an English word that has only one meaning. Synset in ENGLISH WORDNET for such English word has one synset. Polysemic criteria focus on an English word which has multiple meaning. Such an English word has multiple synsets in ENGLISH WORDNET. Whereas, structural criteria emphasize the structural relations between synsets concerning WordNet 1.7. To verify links that constructed using these 13 criteria, a stratified sampling technique has been applied and for each criterion 400 links have been verified manually. The results of verification show that the best criterion has 92% correctness and the lowest correctness is equal to 79.25%.

Montazery et.al [8] propose an automatic method for building a large-scale Persian WordNet based on Princeton WordNet 2.1 (ENGLISH WORDNET). It uses English and Persian corpora. To map Persian words to correct ENGLISH WORDNET synsets it use WordNet similarity. It computes a score for each candidate synset of a given Persian word and select the synset with the maximum score as a link to the Persian word, for each of its translations. They have used the following resources in the process of score calculation: PWN: synset words, synset definition, and hypernym relations, bilingual dictionary (Persian – English), raw Persian text corpus are used for extracting related words of a given Persian word and raw English text corpus for extracting mutual information between English words. In experimental result accuracy of selected links for 500 words, the Precision for unambiguous links is 95.8%, for ambiguous links is 76.4% and for all links is 82.6%.

Khodak *et.al* [4] proposes construction of automated WordNets based upon word-senses joint with readily accessible machine translation tools and distributional representations of sentences. Machine translation is often used for automated WordNets construction to generate a set of candidate synsets for each word w in the target language by getting a set of English translations of w. And use the word embedding-based method for matching words to synsets. They develop a sense clustering scheme to handle polysemy and assign high scores to correct candidate synsets of word w and low scores to incorrect ones based on word sense induction. To evaluate their method they construct 600-word test sets for word-to-synset matching in French and Russian. They produce French and Russian WordNet with F.5-Score 50.3%, F1-Score59.3%, Precision 46.1%, Recall100.0 % &F.5-Score, F1-Score50.2%, Precision59.6%, 45.9%, Recall 100.0% respectively.

3.8 Summary

Several studies have been done to construct WordNet resources. In this chapter, we discussed the different mechanisms of WordNet construction for different languages such as English, Thai, Chinese, Portuguese, Persian, etc. The first English WordNet was carefully constructed manually over many decades.

In general, WordNet can be achieved manually, semi-automated and fully-automated way. The manual method approach is constructing WordNet using hand-crafting. On the other hand, researchers use automatic translation of the Princeton WordNet using Google's online translation resources or using bilingual dictionary. A fully-automated approach for constructing a probabilistic WordNet is also designed for Persian WordNet. But still, it needs external resources like a bilingual dictionary they find all possible definitions of a Persian word in English. However, most of the above-listed WordNet construction methods are depending on an additional resource like a lexical dictionary, English WordNet, or depend on bilingual text corpus between the target language and the English language.

The above methods that we discussed need a resource like monolingual text corpus, bilingual dictionary or effective machine translator, thesaurus etc. The manual construction of WordNet is also time taking and tiresome. The Amharic language has no effective machine translator and resource listed above. So, we propose to construct Amharic WordNet from unlabeled text data by using the advantage of word embedding, which captures the conceptually similar word for a given word.

Chapter 4 Design of Amharic WordNet

4.1 Introduction

In this chapter, we will discuss the design part of the proposed automatic Amharic WordNet construction model using word embedding. Besides, we deal with different text pre-processing tasks. Pre-processing tasks include normalization, stop-word removal, tokenization, and stemming. In addition, we discuss the training activity of the word embedding model and the different mechanisms of extracting WordNet relations from the result of the embedding model. The feature extraction and representation mechanism of distance supervision based synonym relation extraction will also be discussed.

4.2 System Architecture

Figure 4.1 depicts the architecture of automatic construction of Amharic WordNet from unlabeled text data. It has four components namely text-preprocessing, hypernym/hyponym extraction, antonym extraction, and synonym extraction. The next section presents the details of each component.

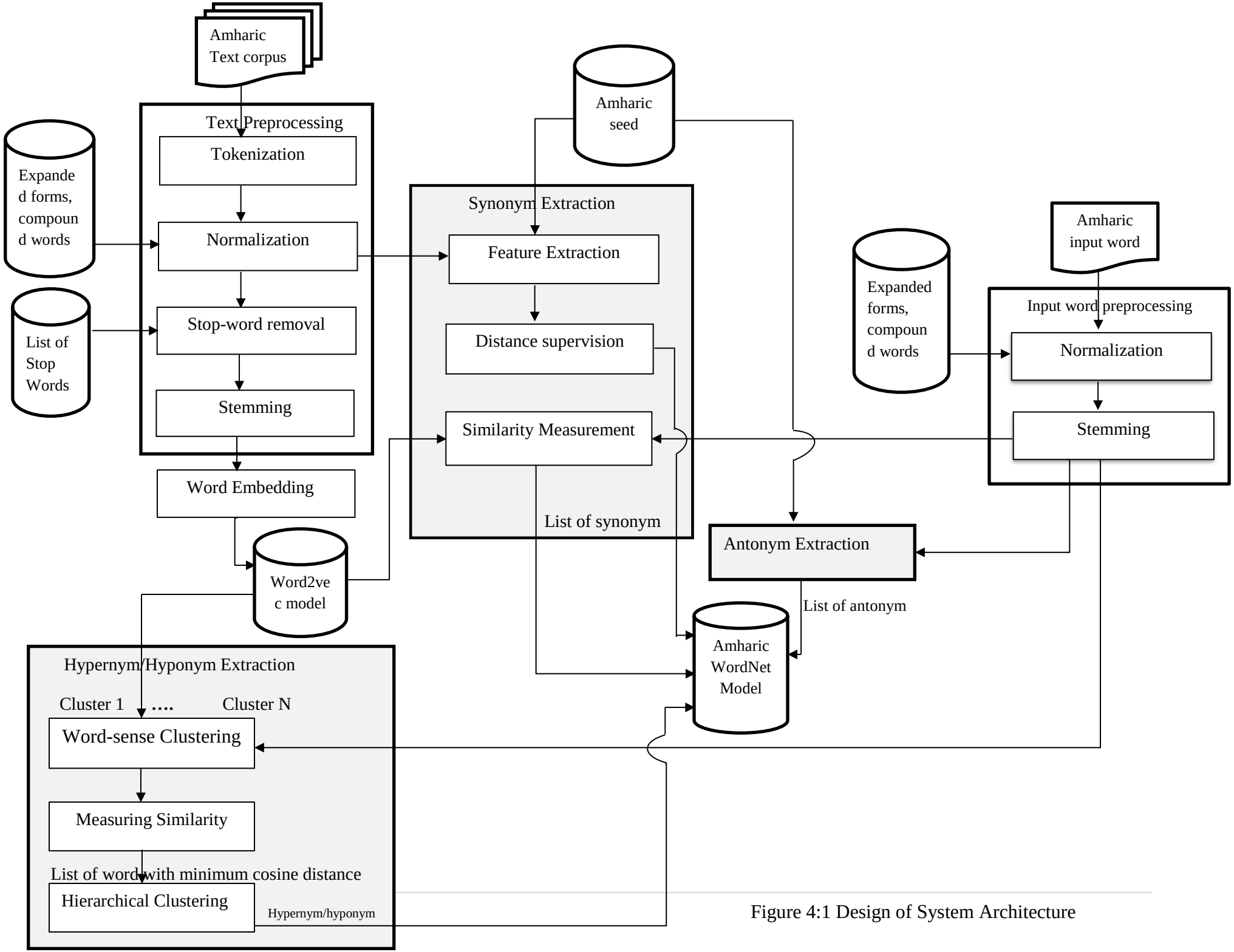

Figure 4:1 Design of System Architecture

4.3 Preprocessing

In this section, the raw data is converted into meaningful data using the process called text preprocessing. The need of preprocessing is to remove raw data that contain unwanted punctuation marks, stop words, numerical value, and special character, and to replace one alphabet in different representations etc. These values can affect the performance as well as the correctness of the model. So, before applying any text modeling or featuring (before training the model) data preparation procedures (like changing text corpus into suitable format) must be done before performing word embedding based on this thesis work. Pre-processing is involved in preparing the input text corpus into a format that is suitable for the word2vec model generation. The pre-processing stage consists of a variety of processes, those are stop word removal, normalization, word segmentation (tokenization), and stemming.

4.3.1 Tokenization

In tokenization (splitting data into a small chunk of words) the whole corpus is split into word level. Tokenization takes the input text supplied from a user and split it into a sequence of tokens, which is the process of breaking a stream of text down into words. And finally, it gives the list of words that is used in the next phase of preprocessing. Due to the model that we use we may also split into character level. In order to generate the word2vec model, the word is an input for the word2vec algorithm which will be extracted from the input text corpus.

In most Latin's language white spaces and other punctuation marks (like question mark {?}) are used as the main approximation of word to word delimiter (boundary markers between sequences of words). Like the other languages, the Amharic language also has its own punctuation marks which separate texts or sentences into a stream of words. Amharic punctuation marks include 'Hulet netb' or colon (፡), 'arat netb '(፨), 'netela serez'(፧), 'derib sereze' (፤), question mark'?' and exclamation mark '! Or ¡' are used as sentence delimeter or as white space. In this thesis work, we take all the punctuation marks as word delimiter but 'arat netb', 'hulet netb', and white space are widely used.

4.3.2 Normalization

Word segmentation (tokenization) splits the give text to words. Normalization is a process that converts a list of words to a more uniform sequence. In Amharic language, three types of

normalization issues arise. The first one is an Amharic word that may be formed from the compound of two Amharic words. For example, if the word "አዲስ" and "አበባ", " ቤተ" and "መፅሐፍ" etc. occur consequently, we have to identify such words and combine them as "አዲስ_አበባ" and "ቤተ_መፅሐፍ". The second one is the identification and replacement of shorter forms of a word. A short form of a word is written using forward slash "/" or period ".". For instance, a word ቀ/ኃ/ሥ or ቀ.ኃ.ሥ is replaced with ቀዳማዊ ኃይለ ሥላሴ. Some of Amharic alphabets have two or more repetitions. So, in order to normalize such issues, we identify and replace Amharic alphabets that have the same use and pronunciation, but that differ in representations of alphabets. The replacement is not random it is made using assigned representative alphabet from a set of similar alphabets. For example, the word "Amhara" can have four representations in Amharic as follow: - "አማራ"," ዐማራ"," ኣማራ" and "ዓማራ". All of these words are differing only by their first characters: "አ","ዐ", "ኣ" and "ዓ". As we have seen those have similar usages and different forms. So to make our word embedding consider as a single word we have to convert the first characters to a single representative character such as "አ".

4.3.3 Stop word removal

Words that need to be cleaned out after or before text preprocessing are called stop words. Stop-words are commonly eliminated from many NLP applications because these words can be distracting the meaning of important words and they need additional memory and are non-informative. There is no common universal list of stop words (maybe context-dependent) for every language. Therefore, it may differ for the different tasks of NLP. But in general, stop words are the most common words (appearing with high frequency in every text) in a language and also they are low information-bearing words. In Amharic language, most of the time, conjunctions, articles, and prepositions occurred frequently and they don't give any useful information in the text. Therefore, we consider these words as stop word for Amharic language. In our task, we are predicting the context of words from a word surrounding it using word embedding. So the reason behind elimination of Amharic stop words before training word2vec model is to potentially improve the power of prediction of word embedding model to, to reduce memory overhead, get a better result, and since we are focusing on the more important terms it reduces noise and false positives, etc.

As we described above, like other languages, Amharic also has low information-bearing words called stop words. For instance, words such as "ና" or "እና", "እስከ", etc. are considered as stop words in this thesis, because they don't have significant discriminating power. So, in this preprocessing step, we discarded the above list of words from input text as these are low information-bearing words. Algorithm 4-1 depicts the steps in the text preprocessing task.

Input: Amharic text document

```
1:    Start
2:    Open Amharic text document from a directory
3:    Read word tokenizers input from disk
4:        Apply tokenization
5:    Read Amharic alphabets form disk
6:        go to line 3
7:        check characters
8:          if true:
9:              replace characters
10 :      else:
11 :          jump
12:   Read compound words form disk
13:       go to line 3
14:       check word sequence
15:         if true:
```

```
16:                         combine it
17:                 else:
18:                         jump
19:     Read stop words from a disk
20:             go to line 3
21:             go to line 5
22:             go to line 12
23:             check words
24:                 if true:
25:                         remove it
26:                 else:
28:                         jump
29:     Else:
30:       End of file
31:     Write the file into the disk
32:     End
```

Output: Preprocessed document

Algorithm 4-1 Algorithm for text pre-processing

The algorithm shown in 4-1 is including preprocessing tasks such as tokenization, normalization and stop word removal.

4.3.4 Stemming

Stemming is the process of changing morphological variants of words into a common form. That means it removes usually affixes from words. For morphologically rich languages like Amharic,

we remove affixes like (prefix, infix, and suffixes) since it is an important preprocessing step in a number of natural language processing (NLP) applications. The assumption behind stemming words is words that have the same root or stem refers to the same concept. It reduces the dimensionality and reduces different forms of a word to their word stems or removing inflectional and derivational affixes. For instance, to show stemming bring words into the same form for the word ሰበረ (Sebere) ፡- (ሰበረች፣ ስለሰበረ፣ ስለሰበሩ፣ እንደሰበረ፣ ከሰበረ፤ ከሰበረች፣ እንደሰበረች፣ ተሰበረ፣ ተሰባሪ፣ ለሰበረ) stem to their stem word "ሰበረ". Here is another example of stemming, for the given phrase "በአለም አቀፍ ደረጃ የተቀሰቀሰው ኮሮና ቫይረስ ከፍተኛ እልቂት እያደረሰ" stemmed as "አለም አቀፍ ደረጃ ተቀሰቀሰ ኮሮና ቫይረስ ከፍተኛ እልቂት ደረሰ".

This helps to reduce the dimension of our text corpus. In addition, it also saves storage and time during training it also increases the accuracy of the training model.

4.4 Word Embedding

After preprocessing tasks of a corpus is done, the next step is to train the mode with word2vec (machine learning algorithms) using the preprocessed text as input to generate word predictions. As we discussed in chapter two, word2vec model can capture contextually similar words for any word that exists in training data. It can create relatedness between concepts. This can be done, based on the assumption of a word and its neighbor deals with the same concept. For any word, its neighbor words provide important information about its meaning. To predict the conceptually related words for each word that exists in the text corpus, training of neural network with a single hidden layer must be done. However, the concepts of words predicted using the weights (the word vectors that the model trying to learn) of the hidden layer. We call this resulting learned vector embedding's. These embeddings are features (conceptually related word) that describe the target words. In the training, the input is each word, along with configurable windows (most of the time 5 to 10 words) in addition to other parameters. Most of word2vec model parameters are used as it is, only some of them needs configuration to get better result.

For instance, below we demonstrate how the training is performed partially. It takes the target word as input and finds its context words. Let us, consider the demonstration with a window size of 3:

መነሻውን ቻይና ያደረገው ገዳይ የሆነው የኮሮና ቫይረስ ወረርሽኝ ከፍተኛ ስጋት እየሆነ መምጣቱን የአለም የጤና ድርጅት አስታወቀ።

The above text after preprocessing becomes this: - መነሻ ቻይና ገዳይ ኮሮና ቫይረስ ወረርሽኝ ስጋት አለም ጤና ድርጅት

Table 4:1 Gives example of target and context words training with window size of 3

With window size 3, pink shows target word and blue-black colors shows context word for each

መነሻ	ቻይና	ገዳይ	ኮሮና	ቫይረስ	ወረርሽኝ	ስጋት	አለም	ጤና	ድርጅት
መነሻ	ቻይና	ገዳይ	ኮሮና	ቫይረስ	ወረርሽኝ	ስጋት	አለም	ጤና	ድርጅት
መነሻ	ቻይና	ገዳይ	ኮሮና	ቫይረስ	ወረርሽኝ	ስጋት	አለም	ጤና	ድርጅት
መነሻ	ቻይና	ገዳይ	ኮሮና	ቫይረስ	ወረርሽኝ	ስጋት	አለም	ጤና	ድርጅት
መነሻ	ቻይና	ገዳይ	ኮሮና	ቫይረስ	ወረርሽኝ	ስጋት	አለም	ጤና	ድርጅት
መነሻ	ቻይና	ገዳይ	ኮሮና	ቫይረስ	ወረርሽኝ	ስጋት	አለም	ጤና	ድርጅት
መነሻ	ቻይና	ገዳይ	ኮሮና	ቫይረስ	ወረርሽኝ	ስጋት	አለም	ጤና	ድርጅት
መነሻ	ቻይና	ገዳይ	ኮሮና	ቫይረስ	ወረርሽኝ	ስጋት	አለም	ጤና	ድርጅት
መነሻ	ቻይና	ገዳይ	ኮሮና	ቫይረስ	ወረርሽኝ	ስጋት	አለም	ጤና	ድርጅት
መነሻ	ቻይና	ገዳይ	ኮሮና	ቫይረስ	ወረርሽኝ	ስጋት	አለም	ጤና	ድርጅት

The neural network creates a dimension with the number of unique words that exist in the text and trains. For instance, for the above example after training the word vector model the above result in Table 4:1 is obtained. It predicts context words for each target word explicitly. The sample in

the table shows how we train the rest of our data. The interpretation is as follow: - for instance, for the target word “ኮሮና” it has the following context words ((“ኮሮና”, “መነሻ”), (“ኮሮና”, “ቻይና”), (“ኮሮና”, “ገዳይ”), (“ኮሮና”, “ቫይረስ”), and (“ኮሮና”, “ወረርሽኝ”). The pink color is the target word whereas the blue-black color in the left and right of the target word is context words.

```
Begin
    Read  preprocessed text file F
    Call  Word Vector Function()
{
      WordEmbeddingConfig()
            setHyperParametrs()
    startTrain()
       if Training_Architecture is 1
            model_type is 'Skip-gram'
       else
            model_type is 'CBOW'
      Word2Vec()
            getHyperParametrs()
            Train()
 }
End
Save trained Model
Stop
Output Feature vector trained in word2vec model
```

Algorithm 4-2 Shows training of word vector Model

Word2vec Trainer

The Word2vec algorithm feeds the pre-processed training text corpus as an input and produces the word vectors (embedding) as output. The algorithm first creates a vocabulary from the training text data and then learns vector representations of the words. It is a mapping of words into vectors of floating numbers using the neural network. The vector representation that is formed using the word2vec algorithm is considered as a feature. Those are represented in semantic vector space, in the form of floating numbers between 0-1. Each unique word in the sample corpus being assigned a corresponding vector in the space called dimensions. A vector space includes hundreds of dimensions. In this space, words that share similar contexts in the corpus are placed close to one another. This is based on the cosine similarity of words. The vector representation result is a model. This model is saved as .bin or .txt or .csv or .mod etc. and is used as a feature to our task. Figure 2:1 and 2:2 shows the architecture of the trainer.

4.5 Amharic WordNet Relation Extraction

WordNet consists of different relationships between words. Among those various relations between words, we will consider relations like synonym, antonym, hypernym, and hyponym in this thesis. The components in sub-sections below presents the mechanisms of WordNet relations extraction.

4.5.1 Hypernym/Hyponym Extraction

Hypernym can be extracted using word embedding, machine translation, pattern-based, and manual method. But pattern-based methods have a problem, in the pattern-based method. Data sparsity issues is the main problem, two words must co-occur in the same context in order for a pattern to be extracted, despite its simplicity and efficiency **[**54**].** Pattern-based approaches suffer from a low coverage issue or low precision issue [28]. For example, let us take the following sentence “some birds recorded in Africa such as Gadwall”, in lexical-pattern based approach such types of sentence may incorrectly detect as Gadwall is of hypernym Africa. So to solve such problems we use word embedding based approach extraction. The approach is applied in trained word2vec model. First, we find top n conceptually related words to identify which word co-occurs together with one word and then find the pattern between two words. The pattern is checked by using the concept of mutual information.

Our method begin from a cluster containing more than one element instead of single element. Besides, instead of combining two clusters in each step, we combine n (the number of initial cluster) number of clusters. And we combine newly constructed one in each cluster. But the combination is not parallel, it is hierarchical. The construction is done based on the cosine difference. Finally, we identify an element having a minimum summation cosine distance and the mutual information within the newly formed cluster.

The assumption to extract hypernym and hyponym relationship of word from word2vec model is: The word2vec model consists of the contexts of each word called synsets. Words with similar contexts tend to occur together is the idea behind in the extraction of context from the text corpus. As we discussed in (section 2.2.2), the semantic field of a hypernym, also known as a superordinate, is broader than that of a hyponym. A hyponym is a word whose meaning is more specific than its hypernym. Because hypernym and hyponym always represent class and sub class

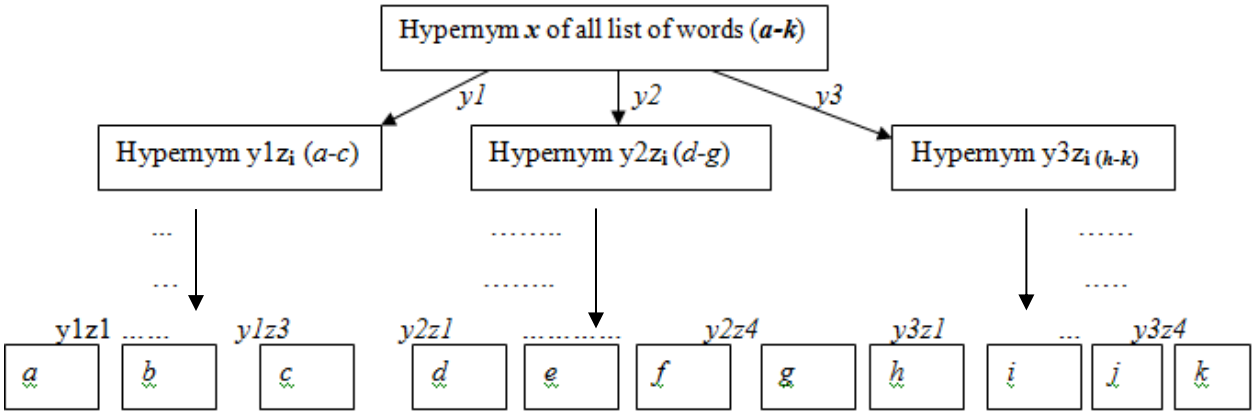

Figure 4:2 shows hypernym hyponym relation

respectively. The class is more general than that of sun-classes. Based on the definition, an element is co-related with others in which that contain common words (mutual information) with a minimum cosine distance. The word having minimum summation cosine distance and that is universal for selected synsets is considered as the hypernym (***x)*** of all other lists of words in the synsets. It is possible to continue the extraction until non-existence of common word between two or more synsets see Figure 4:2 for more detail.

The relationship between hypernym and hyponym is represented as a tree-like structure as we see in Figure 4:2. In order to extract such a relationship, we may think of building a network or tree-like structure in assumption. To extract such relation first get some clusters (more than one cluster) from the trained model. Take any word A, the word must be from the word2vec trained word embedding model. Then find it's A's top n similar (synsets) words using gensim similarity computation class. B, C, D, and F may be the top 4 synsets of word A.

Then discover top n synset list for all list of words (B, C, D, and F) that we get from the word A using gensim. Here A, B, C, D, and F have top n synsets for each. From the definition of the cluster, we can take every synset elements as an individual cluster.

Let us denote clusters of the above synsets as X, Y, Z, W, and V. Those clusters have their elements. Then we combine elements in each synset which have a minimum cosine distance. To construct the hierarchy we find the summation of minimum cosine distance of all newly combined clusters and identifying mutual existence of word.

We find the least common ancestor of all synsets to get the super class.

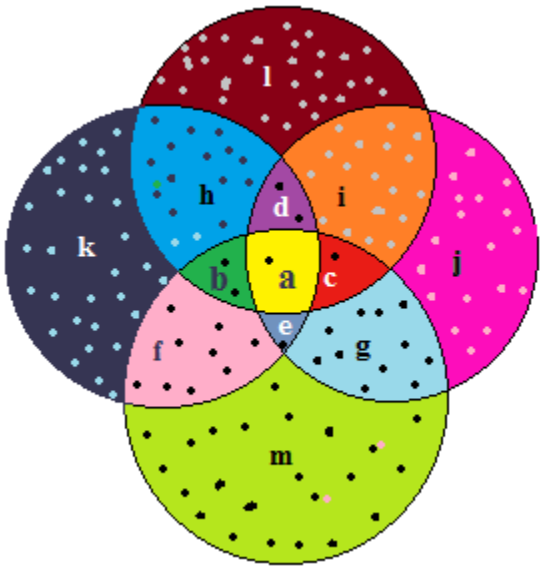

Figure 4:3 Concept of hypernym and hyponym extraction

Since word2vec model capture conceptually related word, more than one word may exist in each cluster. This leads to more than one word considered as hypernym words for a specific list of hypernym words. To handle such a problem, we implement an algorithm that calculates the

summation cosine distance between each word that consists of mutual information of all. Then we built a cosine distance matrix between each point. Then we find the summation minimum cosine distance for each point. Then based on the summation result we decide the hypernym and hyponym words.

Cosine distance between two words calculated as:

$$Cosine - Distance = 1 - Cosine\ similarity$$

Equation 4-1 Equation of cosine distance

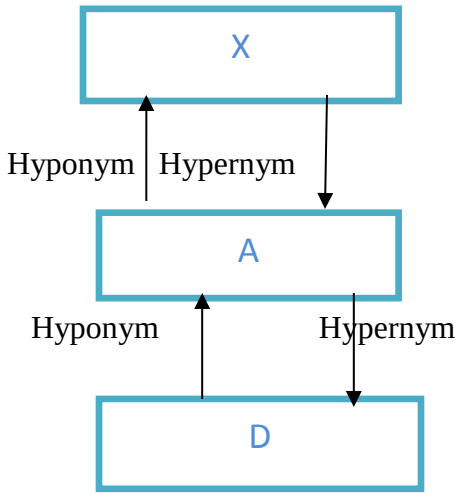

Figure 4:4 Shows Relation between Hypernym Hyponym

Input: Amharic Trained Word vector result
1: Start
2: Open Amharic word2vec embedding from a directory
3: getTopNConceptualRelatedWordsForQueryWord()
4: Resut
5: getTopNConceptualRelatedWordsForWordsInsResult()
6: findCosinedistance()
7: buildDistanceMatricx()

8: findMutualInformation() 9: getWordHavingMinimumSummationCosineDistance() 10: write result of line 9 as hypernym 11: write result of line 4 as hyponym
Stop Output: List of Hypernym-Hyponym

Algorithm 4-3 Hypernym/hyponym relation extraction

4.5.2 Antonym Eextraction

An antonym is also one of the components of WordNet, which is a common lexical relation. It is intuitively clear for humans to define, but not for machines. The need for antonym detection in natural language processing (tasks of understanding language) is: - for paraphrase detection, question answering, and textual inference, etc. [55].

We extract antonym pairs from trained word2vec model using the advantage of word analogy. To do this we extract known antonym pair from English WordNet. Those lists of antonym pairs are English, so we have to translate to the target language (Amharic). Google Translate is a web based application developed by Google which helps to translate text to other languages: it is a free multilingual machine translation service. It converts a word, phrase, paragraph or whole text into a target language that Google implements. Google Translate has an API for software developers. Using this Google translate API we translate each word and its synsets into Amharic language.

We extract antonym words from English WordNet but not those words are much enough. They are limited in number and the translator API is not that much accurate. So we use other techniques to extract antonym pairs. Extraction is done using through applying *word analogies* in a trained Amharic word vector model. Word embedding models (word2vec a model that we use) do not only capture the meaning of words but also can capture multiple different degrees of similarity between words. Since the trained word vector model is a converted high-dimensional vector, semantic patterns can be reproduced using vector simple arithmetic's operations. Thus, we apply the operation in a trained word2vec model to find different word analogies. Word analogy is used to find the relationship among words using a kind of pattern.

Word analogies are done based on the concept of Wa is to Wa* and Wb is to Wb* w stands for word. To exploit this method first each word must be converted into a high-dimensional vector. Then select any two word-pairs from the trained model. Then simply subtract the first vector from the second in the first-word pair (if word pair is a, b ➔ a-b), and add that to the first word in the second word-pair (if c and unknown x) a – b + c. The result is the second word in the second-word pair (a – b + c = unknown). For instance, there is a well-known example of word analogy which is, in a well-trained word2vec model king + woman — man hopefully gives queen.

The analogy may be used to extract different types of relationships. In this thesis work, we apply word analogy for the extraction antonym relation. As we discussed above the method is applied in the high-dimensional vector. In this case, the vector is Amharic word2vec trained model.

Word analogies technique requires some input to create a set of known antonym pairs. So first we should have some list of known pairs of antonym (we call it as seed words or seed set). We already extract antonym pairs using MT, so those list of antonym pairs are considered as seed words. But not all sets can be included in the pair list set. Only those seed sets that exist in the trained vector can be taken. Because to perform the word analogy using arithmetic operation, the seed set that we have chosen must exist in our trained model (in the dataset that we train).

Finally, we extract new pairs of antonym using word2vec king + women - men = queen analogy. We extend this analogy into: if ‘አጭር’ (short) is for ‘ረጅም’ (long) ‘ሞቃት’ (hot) is for ‘ቀዝቃዛ’ (cold). This means ‘አጭር’ + ‘ሞቃት’ - ‘ረጅም’ = ‘ቀዝቃዛ’. Here ‘አጭር’ and ‘ረጅም’ are antonym extracted using MT and ‘ሞቃት’ are query words that we find its antonym pair or pairs. Then we obtain the word ‘ቀዝቃዛ’ based on the above word analogy. So the word ‘ቀዝቃዛ’ can be antonym pair of the ‘ሞቃት’.

Generally, an antonym extraction mechanism is as follows: let us say there are antonym pairs such as X and Y. And also query word Z, which we find its antonym pair. Then we can say X + Z - Y = antonym of (Z). But all the word should exist in word2vec model even if we extract from English WordNet.

Input1: Amharic Word embedding Model
Input2: Sample list of antonym pair

```
Start
    Open Amharic word2vec embedding from a directory
    getAntonymPairFromEnglishWordNet()
        Apply Word analogy Antonym pair 1 + Query Word - Antonym pair 2
            get result
  Write result
  End
Stop
Output: List of antonym pairs
```

Algorithm 4-4 Algorithm for Antonym Extraction

4.5.3 Synonym Extraction

Synonym has a number of definitions. A word having nearly the same or exactly identical meaning as another word in certain contexts is known as synonym word. Synonymy relation is having closely related meanings that exist between words. Usually, a synonym for a collection of terms exists in thesaurus or dictionaries but it is usually incomplete. This is because words do not have a single meaning, it may have many aspects of meanings in different contexts. So it is difficult to judge whether two words are synonym or not [56]. Synonym usually change the meaning a lot based on the same domain. For instance, if someone wants to write a document in a word processor and if he/she wants to replace a word with its synonym, he/she probably wants something closer to the dictionary definition of a synonym. If someone needs a synonym of a word in natural language tasks such as a semantic search engine, he/she definition of synonym can be much broader and include any alternate words that will help him/her get better search results for the query. For example, if you search "what is a banana?" the query may be automatically expanded as "what is fruit". So in this case banana and fruit are synonym, but such a relationship doesn't exist in the dictionary meaning of words. But both words are co-occur in a sentence (banana is a fruit). Word embeddings capture co-occurrence of a word.

In this thesis, we extract both the dictionary meaning as well as contextual meaning on the word. But we use different methods to extract this because of their meaning difference. We apply

distantly supervised relation extraction technique to extract the dictionary meaning of words whereas the contextual meaning words can be done via word embedding. As we have described in section (4.4) word embeddings can train word vectors for the corpus and then find contextual synonym for the current word by using nearest neighbors.

Distantly Supervised relation extraction is a technique which identifies the relationship between two entity mentions in a single sentence. The technique is providing a set of seed sets from an existing Knowledge Base (KB). In the extraction of Amharic synonym words, we use English WordNet as a knowledge base. We extract synonym sets from the KB and translate Using GoogleTranslate API. Next to this, we select and align sentences from our text corpus that matches with each seed pairs (both words of the pair is co-occurring in the sentence), and take those sentences as positive examples for synonym relation type.

Therefore, based on the given examples we automatically extracted features from these sentences (context words). In-order to extract features we hypothesis that every synonym words exist in a text corpus within some kind of pattern. Then we take each seed words (words that are extracted from WordNet) and find them in the text corpus. Then if the word and its synonym exist in a sentence, take co-existence (co-occurrence) words. As a final point to set those words as a feature, compute co-occurrence of term weighting and their precedence order. The words that frequently appear within a specific seed pair with similar precedence order are assigned as context words (features).

Then we extracted synonym words from the text corpus based on the extracted feature. If extracted words exist with other patterns in the text corpus we extract the features and using them get more synsets of synonym. On the other hand, we expand our synonym using the word sense clustering model of word vector. It clusters vectors of contextual similar words across the n-dimensional spaces. The model itself has its own clustering function, using this model we find each terms top n (up to our dimension d) most similar words.

```
Input1: Amharic text document
Input2: Synonym seed words
Start
        getWordNetKnowledgeBase()
```

```
            getSynonymList()
            translateUsingGoogleTranslate()
            pairOfAmharicSynonym
            Open Amharic text from a directory
            findSentenceContainingSynonymPair()
                  if Sentence Containing Synonym pair Of Amharic Synonym
                        extractFeatureFromSentences()
      End
      Write feature
Stop
```

Algorithm 4-5 Algorithm of Synonym feature extraction

Chapter 5 Experiments and Evaluation

5.1 Introduction

In order to validate the proposed Automatic Amharic WordNet construction model, series of experiments were performed in this chapter. The chapter also discusses prototype, data collection and evaluation.

5.2 Experimentation

5.2.1 Data Collection

Since there is no readymade Amharic dataset, the experimentation is beginning with preparing of Amharic dataset that is used for the training and embedding. The dataset that we collect is a text file written (encoded) in Amharic alphabet. The collection of data is done from different media (websites, social media, blogs, etc.) and data sources like magazines, books, and newspapers. The collection is not selective (specific) it includes any written Amharic text document. Nowadays, the web is the most essential source of text documents for different languages. Massive of our Amharic text data is collected from the web. And the rest data is collected from offline sources such as Newspapers: - Amharic reporter ("ሪፖርተር አማርኛ"), Finote Netsanet ("ፍኖተ-ነፃነት"), Magazine: - Feth ("ፍትህ"), etc. Those data collected from online and offline sources are unlabeled texts. But for this study, we also need other types of structured data used as a knowledge base (WordNet).

English WordNet: is used as a structured source of data. The data is available online as a software package. It consists of different relationships like antonym, synonym, hypernym, etc. To access the data, first download the package and install it in Anaconda IDE (integrated development environment). By means of python built-in function extract the data that we want from the knowledge base. The collected dataset consists of 9,492 documents. The data are in different domains. Collected documents with the different domain are shown in the Table 5:1.

Table 5:1 Collected Amharic data and its data sources

Domain	Collected from	Total Document

Religious	አዲስ ዓለም ትርጉም መጽሐፍ ቅዱስ New world Translation of Bible	1268
Politics	Ahmaric reporter “ሪፖረተር አማርኛ”, Finote Netsanet “ፍኖተ-ነፃነት”, Magazine: - Feth “ፍትህ”,	2126
General	ሐተታ ዘዘርአ ያዕቆብ, የውስጥ አርበኞች ገፀባህሪያት , ዳግማዊ አጤ ምኒልክ, Zenedkun Bekele facebook page	3546
Art	ፀጋዬ ገብረ መድህን እግ እንይ	
Health	Ethiopian Ministry of health, Hakim	343
Technology	Ahmaric reporter Fana BC, Amhara mass Media Agency	1349
Sport	Amhara mass Media Agency, Amharic reporter, The wave of Tana, Fasil Kenema,	2250

As shown in table 5:1 we collect data from different data sources in different domain. Total number of documents show the number of collected text files. The size of each text file is different.

5.2.2 Implementation

In this experiment, we use QT Designer, Anaconda software tools for the purpose of programming. Anaconda is a free open-source distribution of programming languages like Python and R, which is used for predictive analytics, machine learning applications, and data science etc. Anaconda

simplifies package management and deployment. From Anaconda supported programming languages and IDEs, we used python and Spider respectively.

Python is an object-oriented programming language that provides rapid application development. It is used for general purpose and high-level programming. Python allows you to develop desktop and web applications. Whereas Spyder is a scientific development environment for python, it includes editing, interactive debugging, testing etc. We performed our training using Gensim (a library that installed in anaconda) open-source library which provides the word2vec class for working with a word2vec model. Natural language Toolkit (NLTK) is the other module that we use in the experiment, it consists of English WordNet corpus.

QT-Designer is a standalone application for designing widget forms for QT application. It allows us to develop a graphical user interface application for python using drag and drop. It can compose and customize the windows or dialogs in a what-you-see-is-what-you-get (WYSIWYG) manner, and test those using different styles and resolutions. We use QT-designer to develop the final Amharic WordNet application.

5.2.3 Data Cleaning

After the data is collected and software tools are specified we performed the pre-processing task. The preprocessing task is started by merging the documents into a single text file in each domain. A single word may have different meaning in different domain. We merged the separate files of each domain in to single file under each category. This help capture the different meanings of a word in different domains.

We have filtered out list stop word around 950 from our collected document and remove it. Some examples of filtered out Amharic stop words are shown in the table 5:2.

Table 5:2 Examples of Amharic stop words

ይህ	ከነ	ያለ	ወደ
ስለ	ተራ	ሙሉ	ጋር
እና	ነው	ግን	ወይም
እንጅ	እንኳ	ናቸው	አዎን
እንዲህ	እነዚህ	ምን	ይኸውም

Since there is no perfect Amharic stemmer some words are incorrectly stemmed. We manually checked the stemmed text data and corrected wrongly stemmed words.

5.2.4 Word Embedding

After preprocessing tasks, the unlabeled free-text document is transformed into numeric values (dense vector representation of words that capture something about their meaning). The main reason is to make a machine learning model to understand and process the natural language. Word embedding captured the contextually related meaning of a word in a document, syntactic, and semantic similarity, and relation with other words. We trained the data using one of word embedding model which is word2vec. And obtained a word2vec using both the continuous bag of word method (CBOW) and the skip-gram method (SG). CBOW is fast and better for capturing semantics in small size documents and also better for capturing frequent words (we use for testing of hypernym algorithm). Skip-gram is better for capturing infrequent words (we use it to test the antonym word analogy model because antonym words do not frequently occur in natural language text corpus). The implementation is in the Gensim Python library. There are different word2vec parameters that we specify in our training. We reconfigured some of them and use the rest default parameters. The parameters that are reconfigure are listed below.

- **Vector Size:** It is the number of dimensions of the embedding, which is the length of the dense vector to represent each word. We implemented with 250 dimensions.
- **Window**: The maximum distance between a target word and context word. For our implementation, we set 4 as the window size.
- **Min_count**: It is the minimum count of words to include the word in a training vocabulary. Words with an occurrence of fewer than the frequency specified will be ignored. In our implementation, we set it 6.
- **Workers**: The number of threads to use while training. This helps to fasten the training algorithm with parallel processing. We have set the multiprocessing.cpu_count() which automatically runs threads parallel with the number of our machine system CPU.
- **Training architecture**: The training algorithm, either CBOW (0) or skip-gram (1). We use both.

Finally, the training result is returned in the file type that we want. It may be text file, or .model or .dat. This file consists of the number of unique vocabulary and dimension number (i.ee 400 300). In addition, there is the vocabulary with its context in real number representation. For instance, a vocabulary "ህመም" represented as follow:

Word vector result example:

ህመም 0.01478592 0.013584773 0.0064129643 -0.07738454 0.004015839 0.09282653 0.09503791 0.0320629 0.0076469155 -0.03237915 -0.05461738 -0.042605624 0.0084219575 0.0058274693 0.033879887 0.044345174 0.08169198 0.11210068 -0.033437673 -0.061426047 0.05849739 .. Up to its dimension.

We interpret this real number result using word2vec built in function. To get top-N most similar words for a given query word we use most_similar() function. Table 5.3 shows top-9 most similar words for a given word. And the real number value shows the cosine similarity between the query word and context word.

Table 5:3 Example of contextual related words

Words	**Top 9 Context words**								
እግር_ኳስ	'ሻምፒዮን', 0.9978640079498291	'ጨዋታ', 0.9972561597824097	'ዋንጫ', 0.99699103832244887	'ቡድን', 0.9967399835586548	'ጂማ_አባጂፋር',0.9924855828285217	'ግጥሚ' 0.9918581247329712	'ክለብ', 0.9912689924240112	'አሸናፊ', 0.9876319169998169	'የጣናው_ሞገድ',0.9799194931983948
ሩጫ	'ኦሎምፒክ',0.9979919791221619	'ኬኒያ', 0.9978920221328735	'ኢትዮጵ ያ',0.9977421760559082	'ወርቅ', 0.9975638985633385	ሜዳሊያ', 0.9974783062934875	'ማራቶን', 0.9974347352981567	'ውድድር', 0.9967647790908813	'አበበ', 0.9953373670578003	'እግር', 0.9950235486030579

ኮሮና	'ህመም', 0.9990732669830322	'በሽታ', 0.9988457560539246	'ገዳይ', 0.9988448619842529	'ትኩሳት', 0.998431801795 9595	'ወረርሽኝ', 0.99838185310363 77	'ኩላሊት', 0.9983801245 68939 2	'ማስነጠስ', 0.99831879138 94653	'ወባ', 0.9982 86485 671997 1	'ቻይና', 0.99823379516 60156
መድሀኒት	'ሚዋጥ', 0.99915301799 77417	'ፓራሲታሞል',0.9 990999 698638 916	'ፈዋሽ', 0.9990819096 56524 7	'ሀኪም', 0.99905 3359031 6772	'አዳኝ', 0.998982 2506904 602	'ሚቀባ', 0.9989 50362 20550 54	'ፓናዶል', 0.9987 812042 236328	'መርፌ', 0.9985 1417541 50391	'አልቤንዳዞል',0.9 966747760772 705
ሞባይል	'ወሬ', 0.9988282918 930054	'ደወለ', 0.9980 853796 005249	'መተግበሪያ',0.99 773091 077804 57	'ቴክኖ', 0.9975 798130 0354	'መልእክት', 0.997365 89193344 12	'አነጋገረ', 0.9968 38092 80395 51	'ሳምሰንግ', 0.9955 3132057 18994	'ጠራ', 0.9918 72549 05700 68	'አወራ', 0.9899569749 832153
ኮምፒውተር	'ላፕቶፕ', 0.9985 353946 685791	'ቶሽባ', 0.9984 853267 669678	'ማሽን', 0.9980 169534 68322 8	'ኤችፒ', 0.99647 748470 3064	'ፓድ', 0.995953 20224761 96	'ታብሌት',0.99 51086 64035 7971	'ራም', 0.9930 776953 697205	'ዲስክ', 0.9928 032755 851746	'ዴስክቶፕ', 0.9927066564 559937

5.2.5 Amharic WordNet Relation Extraction

After training the data using the word2vec model we extract different relations using a different technique. The word2vec implementation is done based on the angular occurrences of words by applying the cosine implementation. Word2vec model has its embedding method that finds the most similar word from a trained vector in an efficient way. We use gensim.models.Word2Vec.most_similar method. This method finds the top-N most similar positive and negative words. If the words similarity is toward negatively it is for negative, and positive words contribute positively towards the similarity. This method computes cosine

similarity between a simple mean of the projection weight vectors of the given words and the vectors for each word in the model. Therefore, using this method we get the most related words for a given query word. After we have top-N related words, we apply different techniques that we discussed in chapter 4 in order to extract different relation of WordNet.

Hypernym and Hyponym Relation

The construction of hypernym/hyponym relation is done using the concept of conditional mutual information (the common word for three or more words) and cosine distance. The extraction of hypernym relation is based on the idea behind agglomerative hierarchical clustering combined with mutual information concept. Partially, the idea of a single link (was not directly applied) method with a little modification is used. It is one of several methods of hierarchical clustering based on grouping similar data objects in bottom-up fashion. The approach combining two clusters at each step that has a minimum Euclidian distance between clusters not yet belonging to the single cluster as each other and builds a distance matrix. In this research work, the approach is not used directly; the difference between the hierarchical clustering and our method is it began from a cluster containing more than one element instead of single element. Besides, instead of combining two clusters in each step, we combine n (the number of initial cluster) number of clusters. First, we apply finding the minimum cosine distance for each cluster and build distance matrix. Then combined newly constructed clusters in every cluster. Finally, we found an element having a minimum summation cosine distance and the mutual information within the newly formed cluster. For example, we consider the top-N similar words as the first cluster. Then we found another top-N similar word for each word existing in the first cluster. Finally, using mutual information concept and the idea of the modified hierarchical clustering we found the hypernym and hyponym words. For instance, the query word ‘ቡድሂዝም’ have the following top 5 similar words with its vector value: [('እስልምና', 0.9985978007316589), ('ሀይማኖት', 0.9985328912734985), ('ሂንዱይዝም', 0.998502254486084), ('ክርስትና', 0.9974943399429321), ('ፓጋን', 0.9954257011413574)]. Then we found those 5 words top 5 similar words. Finally, we found the common word with the highest cosine value (most related word for each cluster) among those all clusters. In Figure 5:1 shows each word cluster and hypernym word. In the Figure 5:1, the first rows show synset (top-N similar words) of the word ‘ቡድሂዝም’. We take it as first cluster one, then the rest five lines differentiate

in square bracket are synset of words that exist in the first cluster. Then the final line shows the word '*ሀይማኖት*' is a hypernym of a word '*ቡድሂዝም*' other words shows in the figure.

```
Top 5 words for a word ቡድሂዝም is:
[('እስልምና', 0.9985978007316589), ('ሀይማኖት', 0.9985328912734985), ('ሂንዱይዝም', 0.998502254486084), ('ክርስትና', 0.9974943399429321),
('ፓጋን', 0.9954257011413574)]
Top 5 words for a word እስልምና is:
[('ሀይማኖት', 0.9991426467895508), ('ሂንዱይዝም', 0.9985983967781067), ('ቡድሂዝም', 0.9985978007316589), ('ክርስትና', 0.9982544183731079),
('ፓጋን', 0.9967718720436096)]


Top 5 words for a word ሀይማኖት is:
[('እስልምና', 0.9991427659988403), ('ክርስትና', 0.9991227984428406), ('ሂንዱይዝም', 0.9988912343978882), ('ቡድሂዝም', 0.9985328912734985),
('ፓጋን', 0.997343897819519)]


Top 5 words for a word ሂንዱይዝም is:
[('ክርስትና', 0.9990326166152954), ('ሀይማኖት', 0.9988912343978882), ('እስልምና', 0.9985984563827515), ('ቡድሂዝም', 0.998502254486084),
('ፓጋን', 0.9979540109634399)]


Top 5 words for a word ክርስትና is:
[('ሀይማኖት', 0.9991227388381958), ('ሂንዱይዝም', 0.9990326166152954), ('ፓጋን', 0.9984548687934875), ('እስልምና', 0.9982544183731079),
('ቡድሂዝም', 0.9974942803382874)]


Top 5 words for a word ፓጋን is:
[('ክርስትና', 0.9984549283981323), ('ሂንዱይዝም', 0.9979539513587952), ('ሀይማኖት', 0.997343897819519), ('እስልምና', 0.9967718720436096),
('ቡድሂዝም', 0.9954257011413574)]


ሀይማኖት is a Hypernym of
['እስልምና', 'ሂንዱይዝም', 'ክርስትና', 'ፓጋን', 'ቡድሂዝም']
```

Figure 5:1 Hypernym relation extractions and different clusters

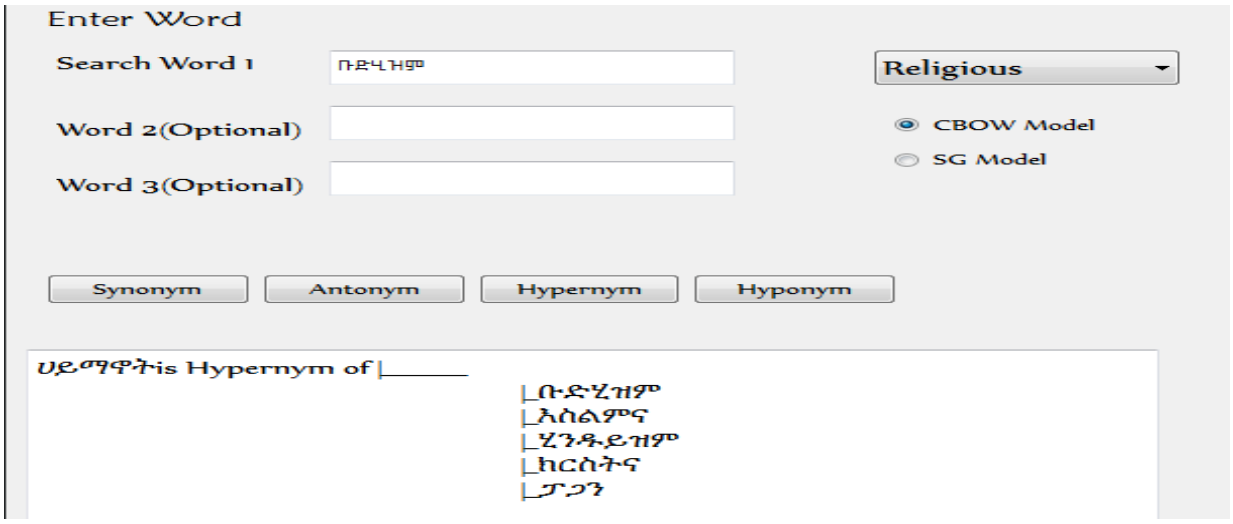

Figure 5:2 Hypernym Relation extractions in graphical user interface

Hyponym relation is the inverse of hypernym relation. Divisive Hierarchical Clustering is also the inverse of agglomerative clustering. Therefore, by applying divisive hierarchical clustering we extract hyponym relation. For example, from the above example for the query word 'ቡድሂዝም' is a hyponym of word 'ሀይማኖት'. So the word 'ቡድሂዝም' is hyponym of 'ሀይማኖት' and since hyponym and hypernym are inverse relationships 'ሀይማኖት' is hypernym of 'ቡድሂዝም'.

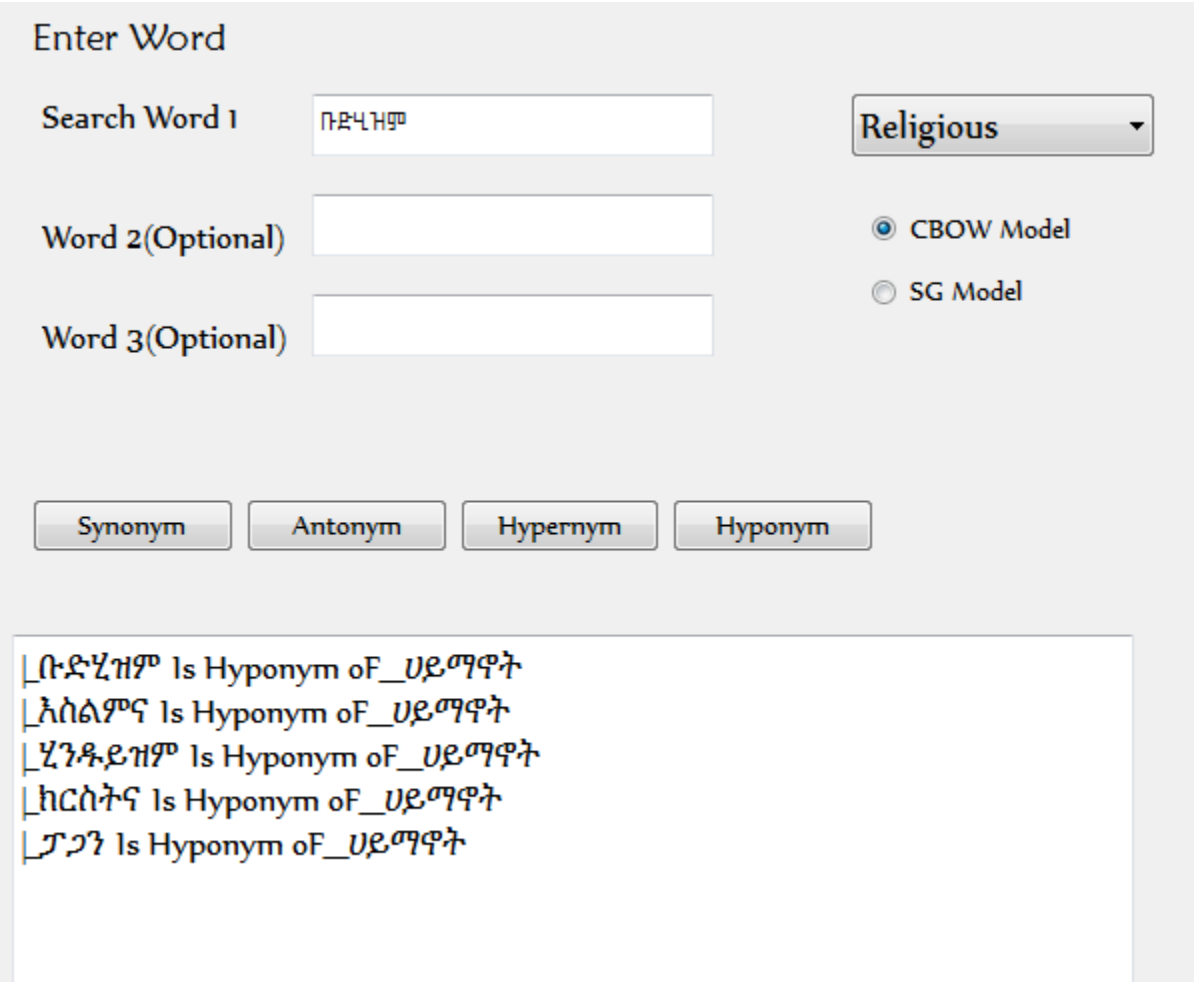

Figure 5:3 Example of hyponym relation

Antonym Relation

Antonym relation is another relation of WordNet. We extract antonym relation using word analogy. As we discuss in chapter 4, word embedding approach is also able to capture multiple different degrees of similarity between words. Semantic patterns can be reproduced using vector

arithmetic. Thus, we could also use word2vec to solve analogies. We use word analogy for antonym extraction. To do this first we get a pair of antonym words from English WordNet (using MT). Then using a pair of words we find antonym of query word. For instance to find antonym of word ‘ውሸት’ we take [‘ትንሽ’ ‘በርካታ’] antonym pair. The antonym of the word ‘ውሸት’ can inquire as using the following analogy ‘ትንሽ’ + ‘ቅርብ’ - ‘በርካታ’ or ‘በርካታ’ + ‘ቅርብ’ -‘ትንሽ’. Antonym pair 1 + query word – antonym pair 2= antonym of a query word is an analogy for antonym relation extraction. We demonstrate the result in Figure 5:4.

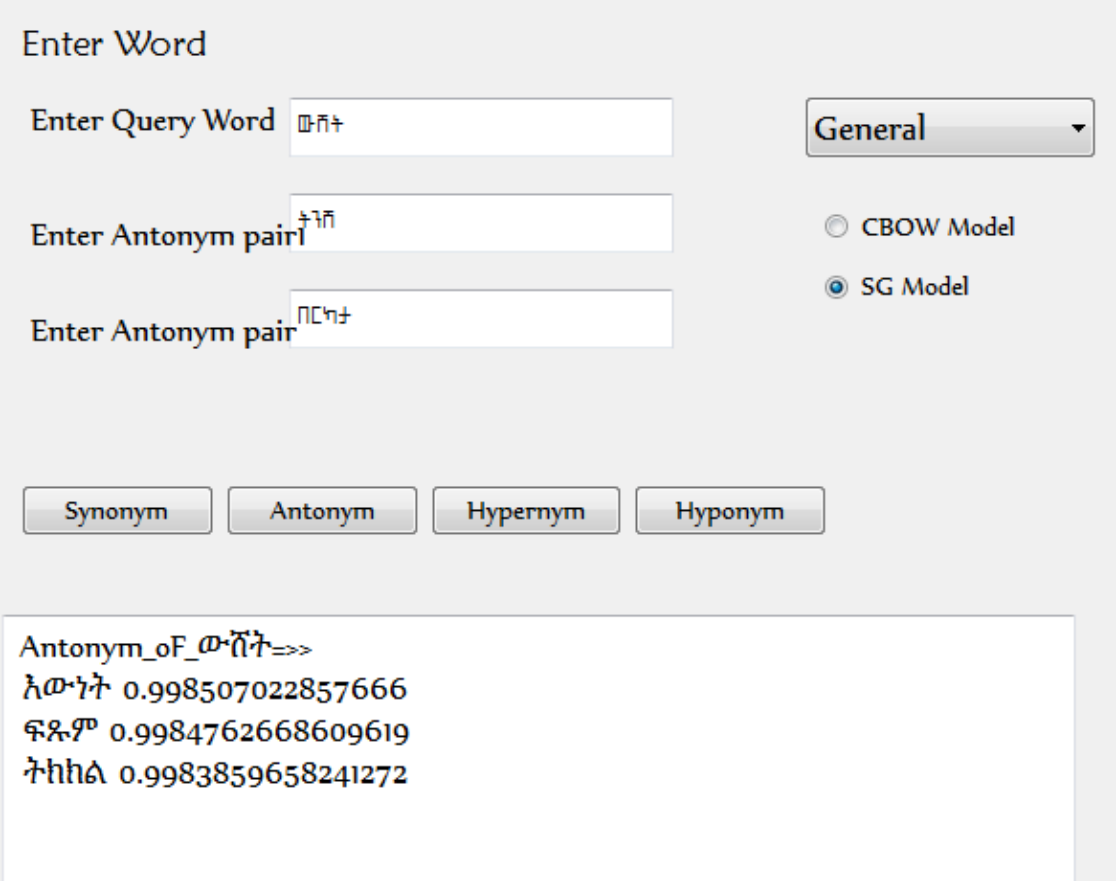

Figure 5:4 Antonym relation extractions

Synonym Relation

Synonym is also a relationship included in WordNet. The relation is extracted using distantly supervised relation extraction technique and word vector model since it has dictionary meaning

and contextual meaning. We extracted a dictionary like the meaning of words using extracting sample synonym directly from the WordNet Knowledgebase. Based on those sample synonym we extracted features. Lastly using this feature, we extracted synonym words as shown in Figure 5:5 and 5:6. And Figure 5:7 shows conceptual synonym extraction using the word2vec model. Using most_similar() method we extract conceptually synonym word for any given query word.

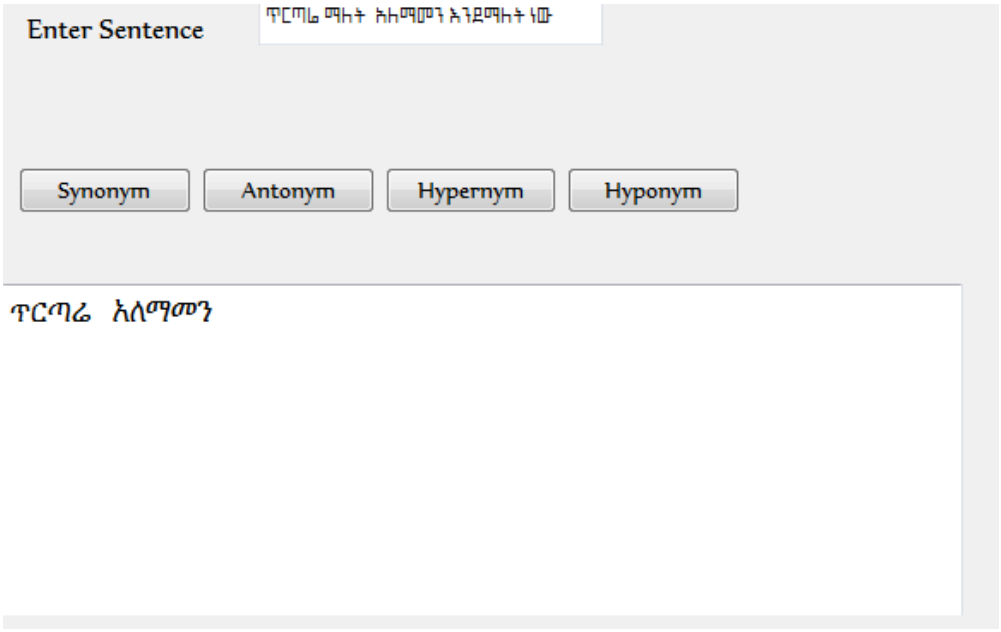

Figure 5:5 Synonym Extraction examples using given sentence

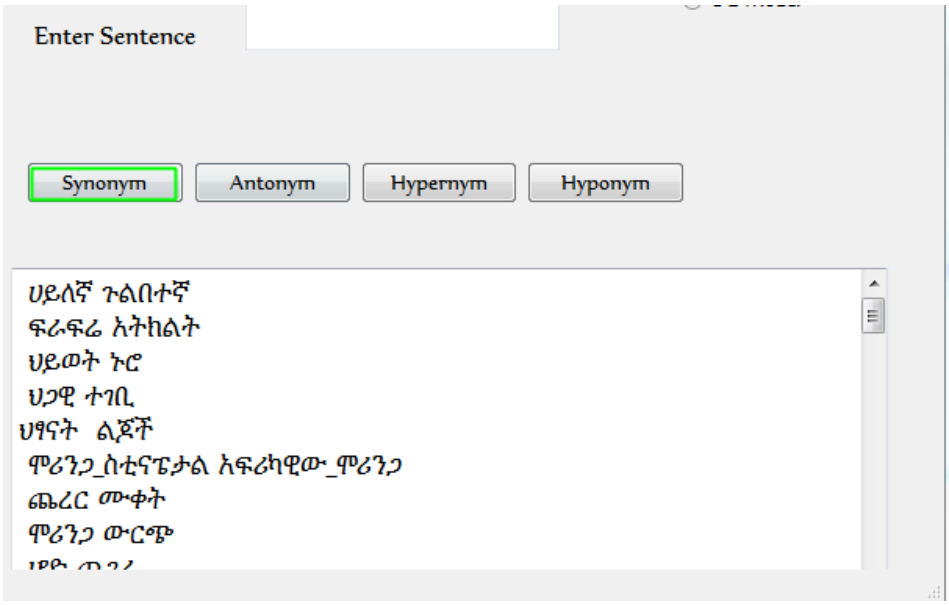

Figure 5:6 Synonym extraction example based on given file

Figure 5:5 and 5:6 shows the result of synonym relation extraction using distant supervised mechanism. Which accept a sentence as input and return synonym word exist in a sentence. In the other hand it take a list of sentence as a file and return all list of synonym words in the file. Figure 5:6 is an example of file input.

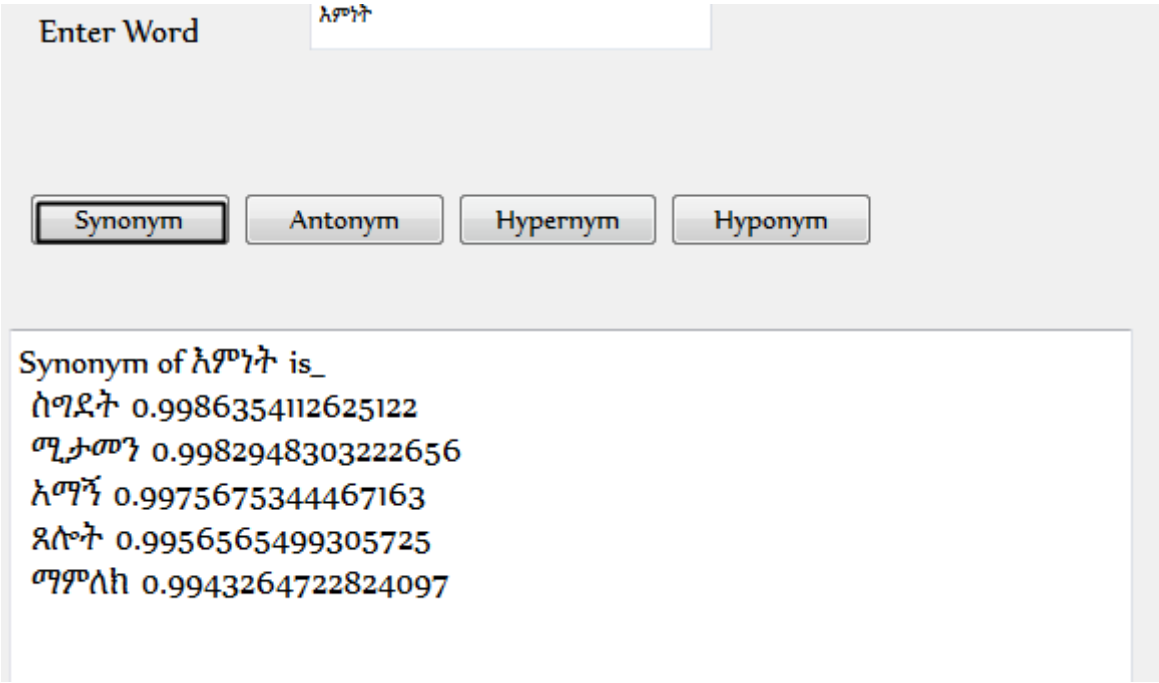

Figure 5:7 Conceptual synonym example using word vector model

5.3 Evaluation

The different relations of Amharic WordNet extracted from the word embedding model are evaluated. The evaluation is based on collected 104 term pairs set (provide pre-collected human-annotated scores of relatedness between term pairs) of different relations. We used word relatedness evaluator (i.e. one type of intrinsic evaluator [57]). It measures the cosine similarity of the embedding for two word pair set with human mean semantic relatedness scores. And it should have a high correlation (Spearman or Pearson) to say well related. To evaluate our system first we prepared questionnaire and built a dataset that consists of relatedness score of 10 language experts. Then we measured the correlation between the human mean semantic relation score provided by the 10 language experts and cosine value calculated by our models. The questionnaire form that we prepared for language experts used to collect relatedness scores to pairs of words, and the

collected experts mean score, as well as the cosine value obtained from the embedding models, is attached as Annex A and Annex B respectively. The correlation calculation is done using Spearman's correlation coefficient. The Spearman rank correlation coefficient, was proposed as a measure of the strength of the associations (how they relate) between two variables [58, 59]. We computed the Spearman's correlation coefficients between each embedding model and human mean score using IBM SPSS statistics software (Version 23). The computation result is shown in Table 5:4.

Table 5:4 Spearman's correlation coefficient result

Correlations				
			WVM cosine value	Human Mean relatedness score
Spearman's rho	WVM cosine value	Correlation Coefficient	1.000	0.790**
		Sig. (2-tailed)	.	0.000
		N	104	104
	Human Mean relatedness score	Correlation Coefficient	0.790**	1.000
		Sig. (2 -tailed)	0.000	.
		N	104	104
**. Correlation is significant at the 0.01 level (2-tailed).				

The computation result in Table 5:4 shows a statistical significance value (ρ=0.0) and Spearman's correlation score (Rs=0.79). The probability ρ value obtained from the calculator is a measure of

how probable or likely is the observed correlation. The value of ρ is in the range 0 and 1 (zero means 0% and 1 means 100%). If the value of ρ is close to zero, it suggests that there is higher correlation between human mean sematic relatedness evaluation and cosine values returned by the system. If ρ-value is close to 1, it signifies no correlation between those two variables that we correlate. So based on the result, we rejected the no correlation null hypothesis and accept the alternative hypothesis that there is a moderate positive correlation between word vector cosine value and human mean semantic relatedness score.

The Spearman correlation coefficient, Rs, can take values from +1 to -1. In this computation, there appears to be a positive correlation Rs value (+0.79). Rs of +1 indicates a perfect association of ranks. Rs value 0 of indicates no association between given compression, and Rs of -1 indicates a perfect negative association. The closer Rs is to zero, the weaker the association between the ranks. Since we achieved Rs of +0.79, it indicates there is a positive correlation of our relatedness assessment (the expert's judgment score and the word vector model value are much more related). As a result, we conclude that the techniques are effective methods to extract WordNet relations. Techniques are methods that we apply in trained word vector model in order to extract WordNet relations.

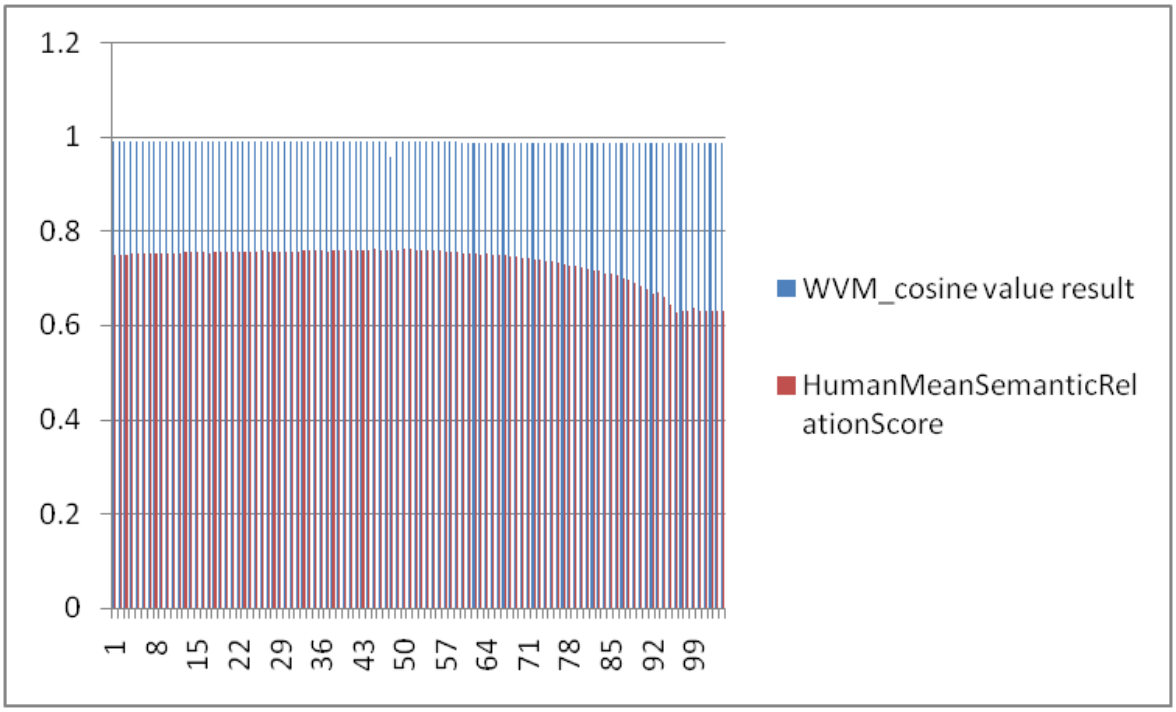

Figure 5:8 Comparison of Human Mean score and word vector cosine score among different relation

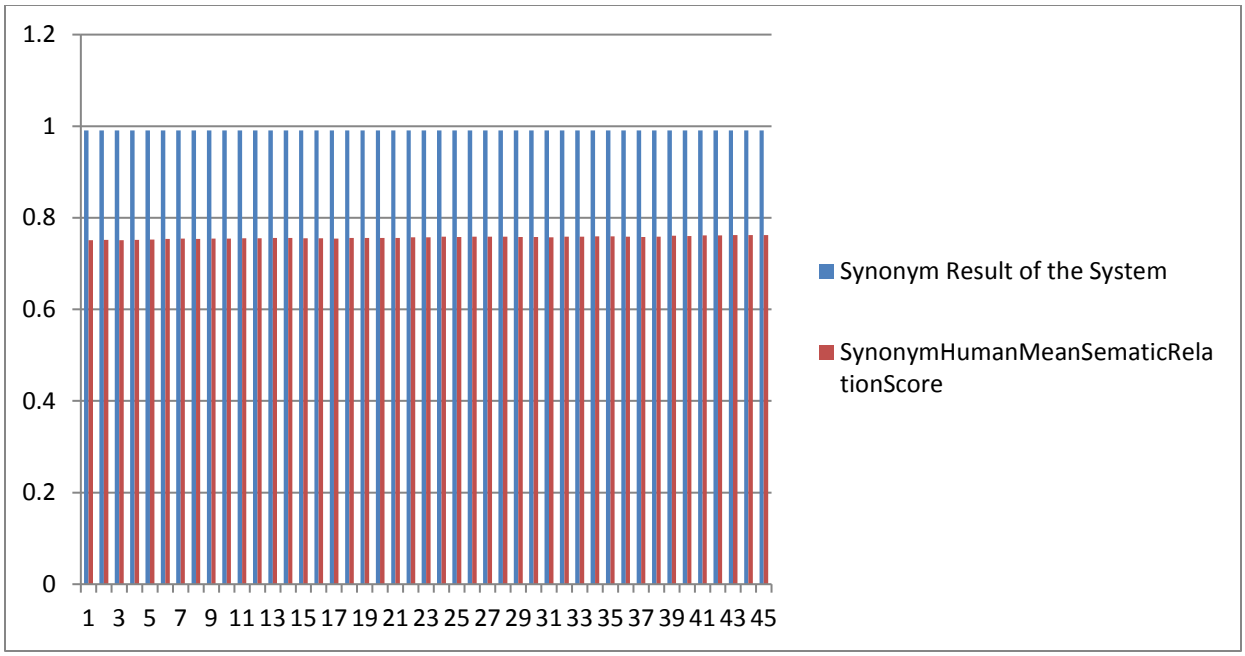

Figure 5:9 Mean Score of Human annotation Vs Synonym model cosine score

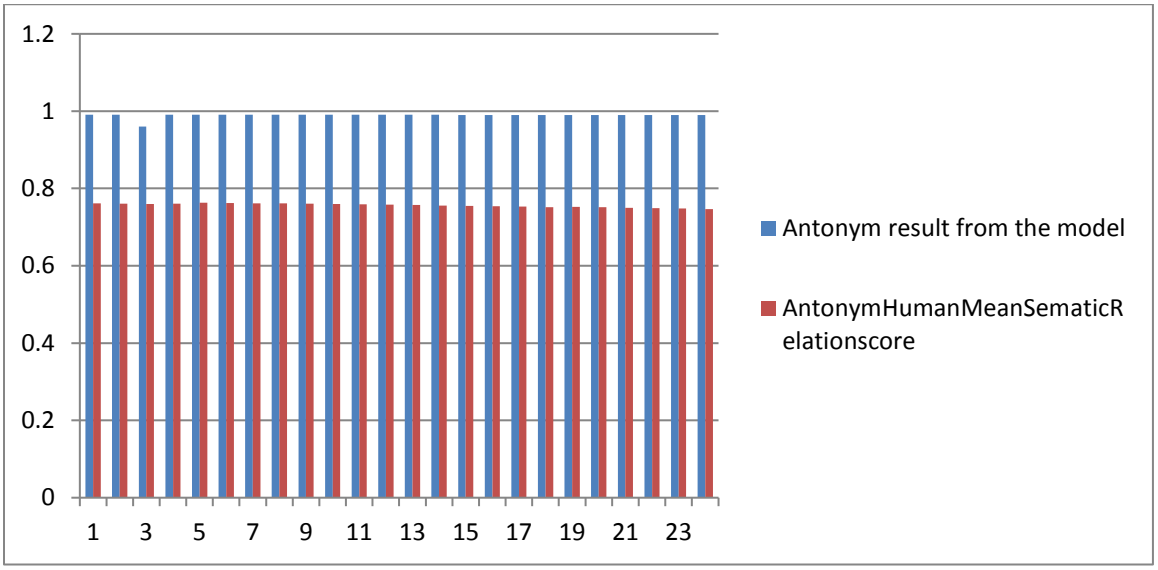

Figure 5:10 Mean score of Human vs. Antonym model cosine Value

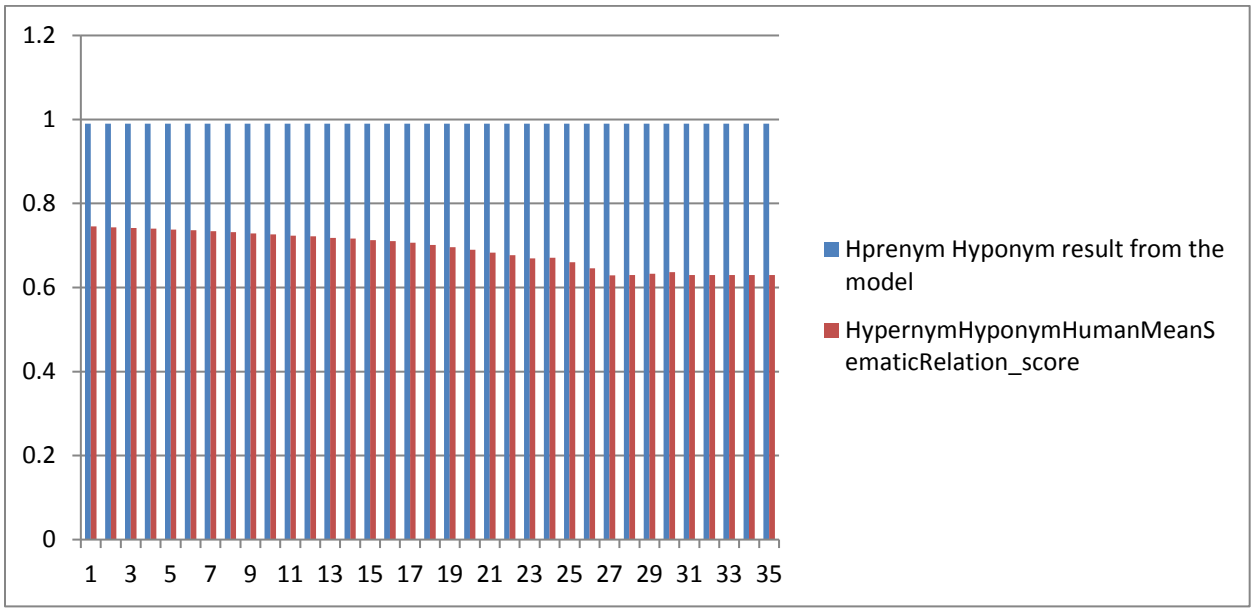

Figure 5:11 Comparison between Human Mean score and Hypernym/Hyponym Model cosine value

In general, the above Figures (Figure 5:9, Figure, 5:10, and Figure 5:11) show the comparison among the cosine values of different relationships and mean scores of Experts' judgement. Whereas figure 5:8 represents the cosine score of all relationships with a human mean score. Figure 5:8 shows the average human semantics relatedness score and the cosine value result returned from our system among various relationships (overall the system). As shown in the Figure 5:8 the average human semantics relatedness score and the cosine value score are highly related. This means the overall system has better accuracy.

Figure 5:9, Figure, 5:10, and Figure 5:11 show the average human semantics relatedness score and cosine similarity of synonym, antonym, and hypernym/hyponym relation respectively. As shown in each Figure the average human relatedness score is between 0.6 and 0.8. Based on the definition of Spearman's correlation coefficient, if there is a small difference among two variables those variables are highly related [58]. So, standing from this definition the score difference of human score and cosine score is small which means the result that we get is better.

5.3.1 Evaluation performance of the System using Precision and Recall

In this section, we measure overall system performance under the prepared dataset. The evaluation is based on a pre-collected 20 query terms of different relationships (synonym, antonym, Hypernym/Hyponym). We used the questionnaire to collect at most top 5 list of words for a specified relation. This collected experts list of words as well as the words returned by the system with the query word are attached as Annex C and Annex D respectively. We collected the dataset by inquiring experts to list at least 1 at most top 5 related words for a given relationship in the questionnaire. We considered the top-listed words in the prepared dataset as relevant words. We evaluated the performance of our systems using commonly used metrics: precision and recall.

We measure the completeness of the query set *R* (recall) and how accurate is the answer set *P* (precision). The expression of recall and precision are as follow:

$$R = \frac{\text{number of extracted word equvalent with experts answer}}{\text{number of expert answer for a given word}}$$

Equation 5-1 Equation of Recall

$$P = \frac{\text{number of extracted words equvalent with experts answer}}{\text{number of all answers retrived from our sysetm}}$$

Equation 5-2 Equation of precision

The detail computation of precision and recall is presented in the Tables 5:5-5:11. The table header consists of abbreviation, the description of those abbreviations is:

- NEA : The number of Expert Answer (taken as relevant)
- RET : Total number of word retrieved by the system
- RET EA: Retrieved words and match with expert answer
- NRETEA : Word not retrieved but exist in expert answer list
- RETNEA : Words retrieved but not in the list of expert answer
- Hyper/hypo : Hypernym hyponym relation type
- WVSyno : Synonym model using Word vector
- DSSyno : Synonym model by distance supervision

– HybridSyno: Hybrid synonym Model

Table 5:5 Precision and recall for synonym word2vec model

Queries Words	Relation Type	NEA	Amharic Synonym evaluation using expert judgment					
			RET	RETEA	NRETEA	RETNEA	R	P
ዘፈን	WVSyno	3	5	2	1	3	0.67	0.4
ጥርጣሬ	WVSyno	3	5	2	1	2	0.67	0.4
መጥፎ	WVSyno	3	5	2	1	3	0.67	0.4
ማብራሪያ	WVSyno	2	5	1	1	4	0.5	0.2
አስነመጂ	WVSyno	3	5	2	1	3	0.67	0.4
ሸሸ	WVSyno	2	5	1	1	4	0.5	0.2
ሹክሹክታ	WVSyno	3	5	1	2	4	0.33	0.2
Average							0.573	0. 314

Table 5:6 Precision and recall for synonym Distance supervision model

Queries Words	Relation Type	NEA	Amharic WordNet evaluation using expert judgment					
			RET	RETEA	NRETEA	RETNEA	R	P
ዘፈን	DSSyno	3	1	1	2	0	0.33	1
ጥርጣሬ	DSSyno	3	1	1	2	0	0.33	1
መጥፎ	DSSyno	3	1	1	1	0	0.33	1

ማብራሪያ	DSSyno	2	1	1	1	0	0.5	1
አስጎመጂ	DSSyno	3	1	1	2	0	0.33	1
ሸሸ	DSSyno	2	1	1	1	0	0.5	1
ሹክሹክታ	DSSyno	3	2	1	2	1	0.33	0.5
Average							**0.379**	**0.929**

Table 5:7 Recall and precision for hybrid model of word vector and distance supervision

Queries Words	Relation Type	NEA	Hybrid Synonym model evaluation using expert judgment					
			RET	RETEA	NRETEA	RETNEA	R	P
ዘፈን	HybridSyno	3	5	2	1	3	0.67	0.4
ጥርጣሬ	HybridSyno	3	5	2	1	2	0.67	0.4
መጥፎ	HybridSyno	3	6	3	1	2	0.67	0.4
ማብራሪያ	HybridSyno	2	6	2	0	4	1	0.33
አስጎመጂ	HybridSyno	3	6	3	0	3	1	0.5
ሸሸ	HybridSyno	2	6	2	0	4	1	0.33
ሹክሹክታ	HybridSyno	3	6	2	1	4	0.667	0.33
Average							0.811	0. 384

Table 5:8 Recall and precision of Amharic antonym relation

Queries Words	Relation Type	NEA	Amharic Antonym relation evaluation using expert judgment					
			RET	RETEA	NRETEA	RETNEA	R	P

ትንሽ	Antonym	3	3	2	1	1	0.67	0.67
ብዛት	Antonym	2	3	1	2	2	0.5	0.33
ውድቀት	Antonym	2	3	2	0	1	1	0.67
ትርፍ	Antonym	1	3	1	0	2	1	0.33
ለያየ	Antonym	3	3	2	1	1	0.67	0.67
መጨምር	Antonym	2	3	2	0	1	1	0.67
ጸመ	Antonym	3	3	2	1	1	0.67	0.67
ው ሸት	Antonym	3	3	2	1	1	0.67	0.67
Average							**0.739**	**0. 569**

Table 5:9 Recall and precision of hypernym hyponym Amharic Relation

Queries Words	**Relation Type**	**NEA**	**Amharic Hypernym hyponym relation evaluation using expert judgment**					
			RET	**RETEA**	**NRETEA**	**RETNEA**	**R**	**P**
አዲስ_ኪዳን	Hyper/hypo	5	5	3	2	2	0.6	0.6
ክርስትና_ሀይማኖት	Hyper/hypo	3	5	3	0	2	1	0.6
አገልጋይ	Hyper/hypo	5	5	4	1	1	0.8	0.8
እጽዋት	Hyper/hypo	5	5	3	2	2	0.6	0.6
ህመም	Hyper/hypo	5	5	4	1	1	0.8	0.8
Average							**0.739**	**0. 539**

Table 5:10 Precision and recall of Amharic WordNet System

Queries Words	Relation Type	NEA	Overall Amharic WordNet evaluation using expert judgment					
			RET	RETEA	NRETEA	RETNEA	R	P
አዲስ_ኪዳን	Hyper/hypo	5	5	3	2	2	0.6	0.6
ክርስትና_ሀይማኖት	Hyper/hypo	3	5	3	0	2	1	0.6
አገልጋይ	Hyper/hypo	5	5	4	1	1	0.8	0.8
እጽዋት	Hyper/hypo	5	5	3	2	2	0.6	0.6
ህመም	Hyper/hypo	5	5	4	1	1	0.8	0.8
ዘፈን	HybridSyno	3	5	2	1	3	0.67	0.4
ጥርጣሬ	HybridSyno	3	5	2	1	2	0.67	0.4
መጥፎ	HybridSyno	3	6	3	1	2	0.67	0.4
ማብራሪያ	HybridSyno	2	6	2	0	4	1	0.33
አስነመጀ	HybridSyno	3	6	3	0	3	1	0.5
ሸሸ	HybridSyno	2	6	2	0	4	1	0.33
ሹክሹክታ	HybridSyno	3	6	2	1	4	0.667	0.33
ትንሽ	Antonym	3	3	2	1	1	0.67	0.67
ብዛት	Antonym	2	3	1	2	2	0.5	0.33
ውድቀት	Antonym	2	3	2	0	1	1	0.67
ትርፍ	Antonym	1	3	1	0	2	1	0.33
ለያየ	Antonym	3	3	2	1	1	0.67	0.67
መጨምር	Antonym	2	3	2	0	1	1	0.67

ጸመ	Antonym	3	3	2	1	1	0.67	0.67
ውሸት	Antonym	3	3	2	1	1	0.67	0.67
Average							**0.783**	**0.539**

The description of the Tables (5:5 – 5:10) interpreted as follow: The first column signifies the word given for expert to write related word under the given relation type (column two). The values in the third column show average number of answers of three experts. The 4th column shows the number of system retrieved answers. Whereas the 5th column shows the number of retrieved answers and exactly match with experts answer. True negatives and false positives are shown in the 6th and 7th column respectively. The last two columns are recall and precision.

Table 5:11 Recall and precision Summary of overall system and different relations

Relation Type	**Recall**	**Precision**
Hypernym/Hyponym	0.739	0.569
Distance Supervision Synonym	0.379	0.929
Word2Vec Synonym	0.573	0.314
Hybrid Synonym	0.811	0.384
Antonym	0.739	0.569
Overall WordNet System	**0.783**	**0.539**

For evaluation purposes, we prepare two datasets: one for evaluation correlation between human mean semantic relation score and word vector model cosine value. The second dataset is used to evaluate the performance of Amharic WordNet system. The second dataset is prepared based on the answers of 3 experts for a given 20 query word containing different relation types. We included the list of words into a dataset that two or more experts have agreed on, for a specific query word. Therefore, the number of list of words for a query word is considered as relevant words (average list of Expert's answers). Based on the data collected we evaluated the performance of our system. As shown in Table 5:11, we obtained a recall of 78.3% and 53.9% precisions for the Amharic WordNet system. We also evaluated the individual components in terms of recall and precision.

. The result is presented in the same Table. As a result, the system shows better performance in both the metrics but much better in the recall.

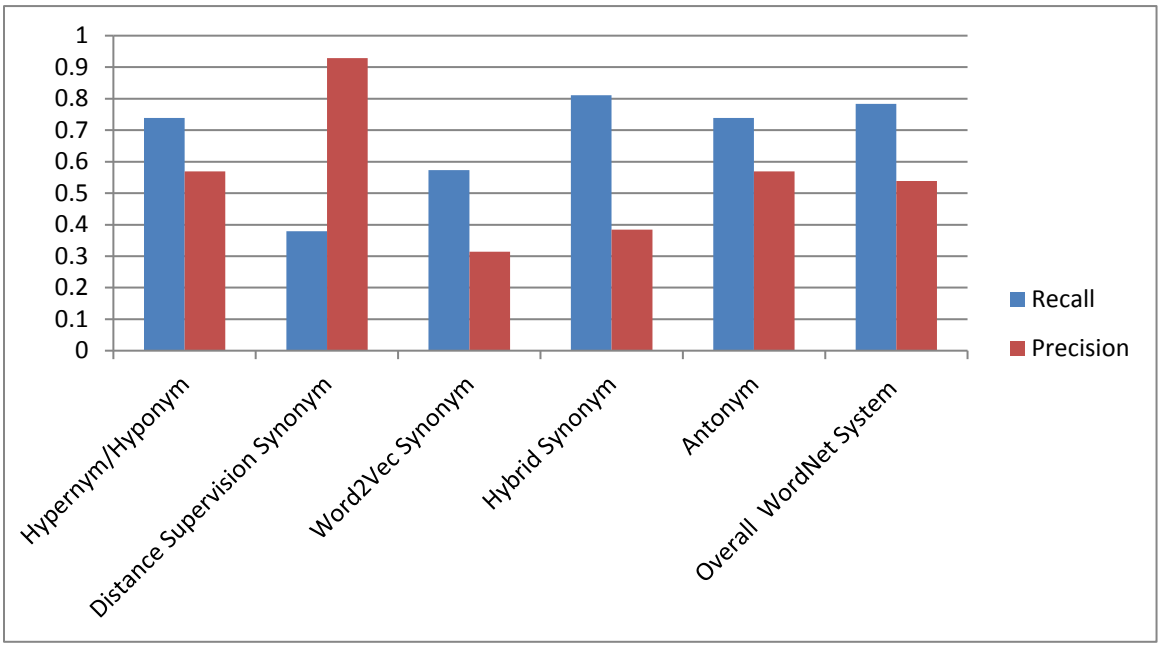

Figure 5:12 Graphical Views of Recall and Precision of each relation and the overall system

5.4 Discussion

We evaluate the system using metrics like Spearman's correlation coefficient, recall, and precision. Using Spearman's correlation coefficient we measure the relatedness score between our model and human relatedness score. As shown in Table 5:4, Spearman's correlation coefficient of Rs=0.79 means there is a high correlation between our Amharic WordNet relation word pairs extracted from our model and human relatedness score. Based on the definition of Spearman correlation, since the correlation coefficient result approaches to one, we can say there is a high correlation. This high correlation shows the correctness of the result. Spearman's correlation coefficient in Table 5:4 is the result of the overall Amharic WordNet model. Statistical significance value P=0 also shows a high correlation. The correlation coefficient and statistical significance value both show the correlation. But the difference is the correlation coefficient shows a high correlation when its value approaches to one whereas statistical significance show high correlation when its value ≈ 0. So, based on the value of both measurements we achieve high correlation.

We also evaluate the Amharic WordNet using precision and recall measuring metrics. In those metrics, we evaluate each relation extractor (hypernym/hyponym, synonym, antonym, and the combined of all relations). Figure 5:12 and Table 5:11 shows the result of precision and recall of each relation and overall system. The precision of distant supervision synonym extraction is higher than other relation extracted by using word embedding. So, there is a low false positive of distant supervision relation extraction. Whereas the recall is very low, the recall of synonym extraction using word embedding is better than distant supervision. That is why we use the hybrid of both approaches to get high precision as well as high recall.

The result of precision and recall for relations like hypernym/hyponym and antonym is presented in Table 5:11. When we compare this result with other languages WordNets it is not a significant improvement. But, the method is very important for languages like Amharic, which have low resources. Amharic language does not have resources like monolingual text corpus, thesaurus, machine-readable bilingual dictionary, effective machine translator, etc. So, WordNet construction using word embedding from unlabeled text data is better for languages like Amharic (those have low resources).

Chapter 6 Conclusion, Contribution and Future Work

6.1 Conclusion

As a result of emerging digital technology, massive amount of structured and unstructured data is produced every day, by the various sources including social media, websites, and news agency. To easily access and use especially unstructured data, the use of natural language applications is inevitable. WordNet is the backbone of many natural language applications including word sense disambiguation, automatic text categorization, information retrieval, and question answering. Even though there are several resources like electronic dictionary and thesauri, WordNet is the largest and wildly used in natural language applications. Since WordNet, is a valuable resource for different applications, creating this resource is very essential. Even if it is possible to construct the resource manually, it is time-consuming and tiresome because the resource consists of different relations. Not only the complication created by the different relation it consists, it also needs huge dataset. Constructing the resource manually from this large dataset is mind-boggling. So as to eliminate such issues, automatic WordNet construction based on the concept of distributional semantics approach is proposed. The distributional semantics approach is the current trend that captures the contextual meaning of words from unstructured text data.

In this research work, we presented automatic Amharic WordNet construction using word embedding. It is implemented using the word2vec model. To achieve the objective of this thesis, first we collected documents containing Amharic text. Then after, we applied the different natural language preprocessing tasks. By using preprocessed text data as input, we implemented the Word2vec model. Word2vec model captures different features of word (consists of a distributional representation of words). Based on those text features, we extracted the different relations of Amharic WordNet. Word2vec returns top-N conceptually related words. Those words are the different relations of the word, but it needs additional implementation to know which type of relationship it is. For instance, hypernym is one of the relation types that exist in WordNet. We extract this relation based on mutual information concept. Using agglomerative hierarchical clustering combined with mutual information concept, we find the hypernym word for any given word. If a word shares information among different clusters, that word is a common word for those clusters. Since hypernym is a more general word and shares its information for its hypernym, we consider the common word as a hypernym of retrieved clusters. Then based on the cosine value of

word vector, we compute and get the hypernym word within the rest hyponym words that exist in the given cluster. Since the representation includes a different feature of the text, we can also extract antonym relation based on word analogy. Word analogy captures different patterns of words, from those patterns antonym pattern is one. A synonym is also another relation of WordNet among the different relations. For this relation, distance supervision method is used in addition to word embedding (word2vec model), which means it is implemented via both approaches. For that reason, the relationship is extracted based on both of the approaches, and the result is aggregated together. Both of the approaches use distributional semantics concept (words occur in similar context have similar meaning). But distance supervision is based on a supervised learning mechanism (use some example of synonym as a seed word). It captures a feature of synonym from the given example and extracts a new list of synonym based on extracted features.

The experimental result in this study showed a considerable performance on Amharic specified WordNet relations (Hypernym, Hyponym, Synonym, and Antonym) extraction. A recall measure of 74.3% recorded during testing proves the high performance of our model. We conclude that to the best of our knowledge, this is a novel research work on Amharic WordNet construction, and achieve better result both in the completeness as well as the correctness of the implemented model. To the best of our knowledge this is the first research in extraction of WordNet relation from unlabeled data. And this is very help full for languages which does not have sufficient resource (bilingual dictionary, monolingual text corpus, thesaurus etc.). The evaluation result shows false positive below 50 %. This means the correctness of our model is better.

6.2 Contribution of the study

The major contributions of this study include:

- Designing a novel Amharic WordNet model.
- A generic model is designed for unsupervised WordNet relations extraction.
- Algorithm which can extract hypernym/hyponym from word embedding result by taking the advantage of mutual information concept and agglomerative hierarchical clustering concept.
- Algorithm that extracts antonym relation of any language based on word analogy.

6.3 Future Work

The system designed in this study is Automatic Amharic WordNet construction using word embedding. WordNet has a number of relation types. However, this study includes only four main relations that are commonly used in different natural language processing applications. In addition, words the so-called **Polysemy** (is the association of one word with two or more distinct meanings) are not handled in this study. So we recommend further research in this study:

- Using sense embedding (sense2vec model) and tagged dataset it is possible to handle polysemy of words and context-sensitive words. This enhances the WordNet, because it captures the different sense of the word.
- Expanding WordNet relations by adding additional relations.
- Extracting WordNet relation using hybrid approach (machine translation and word embedding) enhances the Amharic WordNet.

References

[1] G.G. Chowdhury, Natural language processing, Annu. Rev. Inf. Sci. Technol. 37 (1), 51–89 2003.

[2] George A. Miller, Richard Beckwith, Christiane Fellbaum, WordNet: A Lexical Database for English. Communications of the ACM Vol. 38, No. 11: 39-41, 1995.

[3] Tessema Mindaye, Teshome Kassie, The Need for Amharic WordNet. The 5th International Conference of the Global WordNet Association (GWC-2010) 31st Jan. – 4th Feb., 2010 Mumbai, India.

[4] Mikhail Khodak, Andrej Risteski, Christiane Fellbaum and Sanjeev Arora, 2017, Automated WordNet Construction Using Word Embeddings, In Proc. Workshop on Sense, Concept and Entity Representations and their Applications.

[5] Sathapornrungkij, P., Pluempitiwiriyawej, C. Construction of Thai WordNet lexical database from machine readable dictionaries. In Conference Proceedings: the tenth Machine Translation Summit, Thailand, pages 87-92, 2005.

[6] Fatemeh Torabi, Robert Zinkov, Michael N, Querying Word Embeddings for Similarity and Relatedness, Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1.

[7] Guseong-dong, Yuseong-gu Taejon, A Korean Noun Semantic Hierarchy (Wordnet) Construction, 2002.

[8] Mortaza Montazery, Heshaam Faili, Automatic Persian WordNet Construction, in Proceedings of the 23rd International Conference on Computational Linguistics: Posters, 2010, pp. 846–850.

[9] Marrakech, Morocco, Shamsfard, M. Towards Semi Automatic Construction of a Lexical Ontology for Persian, 2009, In Proceedings of the Sixth International Language Resources and Evaluation (LREC'08), Marrakech, Morocco.

[10] http://amharic.com/wp/amharic/. Friday, Nov, 1, 2018.

[11] Woldeyohannis Michael. Amharic-English Speech Translation in Tourism Domain.

[12] Yesuf, Seid Hassen, and Yaregal Assabie. "Amharic Word Sense Disambiguation Using Wordnet." (2015).

[13] Stanojević, Maja (2009), *"Cognitive synonymy: a general overview"* , Facta Universitatis, Linguistics and Literature series, **7** (2): 193–200.

[14] Palmer, F. R. Semantics: A New Outline, CUP, Cambridge. 1977.

[15] Cruse, D. A. Meaning in Language: An Introduction to Semantics and Pragmatics, OUP, Oxford, 2000.

[16] Doddington, George. Automatic Evaluation of Machine Translation Quality Using Ngram Co-Occurrence Statistics. In *Proceedings of the 2nd International Conference on Human Language Technologies (HLT)*, pages 138–145, 2002.

[17] Gupta, Rohit, Constantin Orăsan, and Josef van Genabith. ReVal: A Simple and Effective Machine Translation Evaluation Metric Based on Recurrent Neural Networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Lisbon, Portugal, 2015b.

[18] Denkowski, Michael and Alon Lavie. Meteor Universal: Language Specific Translation Evaluation for Any Target Language. In *Proceedings of the EACL 2014 Workshop on Statistical Machine Translation*, 2014.

[19] Vylomova, Ekaterina, et al. "Take and took, gaggle and goose, book and read: Evaluating the utility of vector differences for lexical relation learning." *arXiv preprint arXiv:1509.01692* (2015).

[20] Wu, Hua, and Ming Zhou. "Optimizing synonym extraction using monolingual and bilingual resources." *Proceedings of the second international workshop on Paraphrasing-Volume 16*. Association for Computational Linguistics, 2003.

[21] Hazem, Amir, and Béatrice Daille. "Word Embedding Approach for Synonym Extraction of Multi-Word Terms." Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018). 2018.

[22] *St*ede, Manfred (June 2000). *"The hyperonym problem revisited"* . The hyperonym problem revisited: Conceptual and lexical hierarchies in language generation. *Association for Computational Linguistics. January 2014.*

[23] Hearst, Marti A. "Automated discovery of WordNet relations." *WordNet: an electronic lexical database* 2 (1998).

[24] Morin, Emmanuel, and Christian Jacquemin. "Automatic acquisition and expansion of hypernym links." *Computers and the Humanities* 38.4 (2004): 363-396.

[25] Bishop, Christopher M. *Pattern recognition and machine learning*. springer, 2006.

[26] Snow, Rion, Daniel Jurafsky, and Andrew Y. Ng. "Learning syntactic patterns for automatic hypernym discovery." *Advances in neural information processing systems*. 2005.

[27] Lin, Dekang, and Patrick Pantel. "Discovery of inference rules for question-answering." *Natural Language Engineering* 7.4 (2001): 343-360.

[28] Snow, Rion, Daniel Jurafsky, and Andrew Y. Ng. "Semantic taxonomy induction from heterogenous evidence." *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2006.

[29] Sheena, N., Smitha M. Jasmine, and Shelbi Joseph. "Automatic extraction of hypernym & meronym relations in english sentences using dependency parser." *Procedia Computer Science* 93 (2016): 539-546.

[30] Fu, Ruiji, et al. "Learning semantic hierarchies via word embedding." Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vol. 1. 2014.

[31] Baisa, Vít, and Vít Suchomel. "Corpus Based Extraction of Hypernym." *RASLAN*. 2015. p, 69-74.

[32] Finegan, Edward. Language: Its structure and use. Cengage Learning, 2014.

[33] Thalenberg, Bruna. “Distinguishing Antonym from Synonym in Vector Space Models of Semantics.” (2016).

[34] Scheible, Silke, Sabine Schulte Im Walde, and Sylvia Springorum. "Uncovering distributional differences between synonym and antonym in a word space model." *Proceedings of the Sixth International Joint Conference on Natural Language Processing*. 2013.

[35] Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781*(2013).

[36]. https://jaxenter.com/deep-learning-search-word2vec-147782.html Friday, Oct, 12, 2019.

[37] Luis Espinosa-Anke, Steven Schockaert, SeVeN: **A**ugmenting Word Embeddings with Unsupervised Relation Vectors. Proceedings of the 27th International Conference on Computational Linguistics, pages 2653–2665 Santa Fe, New Mexico, USA, August 20-26, 2018

[38] Jon Ezeiza Alvarez. A review of word embedding and document similarity algorithms applied to academic text. October 22, 2017.

[39] Radim Rehuek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, 5 2010. ELRA

[40] Edgar Altszyler, Mariano Sigman, Sidarta Ribeiro and Diego Fernández Slezak. Comparative study of LSA vs Word2vec embeddings in small corpora: a case study in dreams database. 11 Apr 2017.

[41] Socher, Richard, and Richard Socher Mundra. "CS 224D: Deep Learning for NLP1." (2016).

[42] Lebret, Rémi Philippe. *Word Embeddings for Natural Language Processing*. No. THESIS. EPFL, 2016.

[43] Kassa Gobena, Markos. "Implementing an open source amharic resource grammar in GF." (2011). Chalmers University of Technology University of Gothenburg Department of Computer Science and Engineering Göteborg, Sweden, November 2010.

[44] Michael Steinbach, George Karypis and Vipin Kumar, "A Comparison of Document Clustering Techniques", Proceedings of the International KDD Workshop on Text Mining, June 2000.

[45] Hsin-Hsi Chen, Chi-Ching Lin and Wen-Cheng Lin. Construction of a Chinese-English WordNet and Its Application to CLIR.

[46] Marcelo M. Gomes1, Walber A. R. Beltrame2, Davidson Cury. Automatic Construction of Brazilian Portuguese WordNet.

[47] Khang Nhut Lam, Feras A. Tarouti, and Jugal Kalita. Automatically constructing wordnet synsets. In *52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014), Baltimore, USA, June.*

[48] Fadaee, Marzieh, Hamidreza Ghader, Heshaam Faili, and Azadeh Shakery. Automatic WordNet construction using markov chain monte carlo. *Polibits* 47 (2013): 13-22.

[49] Phyue, Soe Lai. "Construction of Myanmar WordNet lexical database." 2011 IEEE Student Conference on Research and Development. IEEE, 2011.

[50] Feras Al Tarouti, Jugal Kalita. Enhancing Automatic Wordnet Construction Using Word Embeddings.

[51] Francis Bond, Hitoshi Isahara, Sanae Fujita. Enhancing the Japanese WordNet.

[52] Zahra Mousavi , Heshaam Faili. Persian Wordnet Construction using Supervised Learning.

[53] Marcelo M. Gomes1, Walber A. R. Beltrame2, Davidson Cury. Automatic Construction of Brazilian Portuguese WordNet.

[54] Zheng Yu, Haixun Wang, Xuemin Lin, and Min Wang. Learning term embeddings for hypernymy identification. In *IJCAI*, pages 1390–1397, 2015.

[55] Silveira, Natalia. Learning to identify antonym. 2013

[56] Li Zhang, Jun Li, Chao Wang. Automatic Synonym Extraction Using Word2Vec and Spectral Clustering. Proceedings of the 36th Chinese Control Conference July 26-28, 2017, Dalian, China.

[57] Tobias Schnabel, Igor Labutov, David Mimno, "Evaluation Methods for Unsupervised Word Embeddings," in Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 298–307, Lisbon, Portugal, 17-21 September 2015. c 2015 Association for Computational Linguistics.

[58] Daniel, W. W. (1990). Spearman Rank Correlation Coefficient. Applied Nonparametric Statistics (2nd ed., pp. 358-365). Boston, MA: PWS-Kent.

[59] Budanitsky, Alexander, and Graeme Hirst. "Evaluating wordnet-based measures of lexical semantic relatedness." *Computational Linguistics* 32.1 (2006): 13-47.

Annexes

Annex A: Questionnaire form offered to Language Experts to Estimate Semantic Relatedness between Pairs of Words for a Given Relation Type

We kindly ask you to assist us in a psycholinguistic experiment, aimed you to estimate the relatedness score for 104 pair of Amharic language word for a specified relation. Based on the estimation word semantic relatedness evaluator measures how well human perceived relatedness is captured by our WordNet model; it correlate the distance between word vectors and human perceived semantic relatedness.

Below is a list of pairs of words. For each pair, please assign a numerical relatedness score between 0 and 10 (0 = words are totally unrelated, 10 = words are very closely related). The relation type *Hyper/hypo* stands for Hypernym/hyponym relation between pair of word.

Specific instructions:

- The questionnaire starts on the next page.
- Please fill in your full name at the beginning of the questionnaire.
- Please fill in the scores in the appropriate column of the table.
- Please DO NOT consults your friends on assigning the relatedness scores – it is highly important that the scores you assign be independent of someone else's assessment.

Full Name: _____

NO	Relation Type	Word pair 1	Word pair 2	Semitic relatedness score between word pair1 and2 out of 10.
1	Synonym	ጥምቀት	መስቀል	
2	Synonym	ጥምቀት	በአል	
3	Synonym	ጥምቀት	ልደታ	
4	Synonym	ጥምቀት	ገና	
5	Synonym	ጥምቀት	ኢድ	
6	Synonym	ፊልም	ሲኒማ	
7	Synonym	ፊልም	ሚታይ	
8	Synonym	ፊልም	ተዋናኝ	
9	Synonym	ፊልም	ምስል	
10	Synonym	ፊልም	አክተር	
11	Synonym	ሞባይል	ወሬ	
12	Synonym	ሞባይል	ደወለ	
13	Synonym	ሞባይል	መተግበሪያ	
14	Synonym	ሞባይል	ቴክኖ	
15	Synonym	ሞባይል	መልእክት	
16	Synonym	ታብሌት	ፓድ	
17	Synonym	ታብሌት	ኤችፒ	
18	Synonym	ታብሌት	ዲስክ	
19	Synonym	ታብሌት	ራም	
20	Synonym	ታብሌት	ማሽን	
21	Synonym	ዊነዶው	ባንክ	
22	Synonym	ዊነዶው	ኦፊስ	
23	Synonym	ዊነዶው	ኢንሹራንስ	
24	Synonym	ዊነዶው	ኡቡንቱ	
25	Synonym	ዊነዶው	ማክ	
26	Synonym	ኮምፒውተር	ላፕቶፕ	
27	Synonym	ኮምፒውተር	ቶሸባ	
28	Synonym	ኮምፒውተር	ማሽን	
29	Synonym	ኮምፒውተር	ኤችፒ	
30	Synonym	ኮምፒውተር	ፓድ	
31	Synonym	ዘፈን	ሰማ	
32	Synonym	ዘፈን	አደመጠ	
33	Synonym	ዘፈን	ቴዲ	
34	Synonym	ዘፈን	ጥላሁን	
35	Synonym	ዘፈን	ዘፋኝ	
36	Synonym	ደራሲ	ተራኪ	
37	Synonym	ደራሲ	አዘጋጅ	

38	Synonym	ደራሲ	ድምጻዊ
39	Synonym	ደራሲ	አርቲስት
40	Synonym	ደራሲ	ኤዲተር
41	Synonym	ማሲነቆ	ዋሽንት
42	Synonym	ማሲነቆ	በገና
43	Synonym	ማሲነቆ	ክራር
44	Synonym	ማሲነቆ	ከበሮ
45	Synonym	ማሲነቆ	መሳሪያ
46	Antonym	ትንሽ	በቂ
47	Antonym	ትንሽ	ብዙ
48	Antonym	ትንሽ	ሆስፒታል
49	Antonym	ብዛት	ትውልድ
50	Antonym	ብዛት	እጥረት
51	Antonym	ውድቀት	ግብ
52	Antonym	ውድቀት	ተሳካ
53	Antonym	ውድቀት	ስኬት
54	Antonym	ትርፍ	ባዶ
55	Antonym	ትርፍ	ኪሳራ
56	Antonym	ለያየ	ጨመረ
57	Antonym	ለያየ	መቀላቀል
58	Antonym	ለያየ	ማድረግ
59	Antonym	መጨምር	አነሳ
60	Antonym	መጨምር	ማንሳት
61	Antonym	መጨምር	መቀነስ
62	Antonym	ጸመ	በላ
63	Antonym	ጸመ	ማኘክ
64	Antonym	ጸመ	ምግብ
65	Antonym	ውሸት	እውነት
66	Antonym	ውሸት	ትክክል
67	Antonym	ውሸት	ፍጹም
68	Antonym	ሩቅ	ከታች
69	Antonym	ሩቅ	ቅርብ
70	Hyper/hypo	አዲስ_ኪዳን	ፊሊጵስዩስ
71	Hyper/hypo	አዲስ_ኪዳን	እብራውያን
72	Hyper/hypo	አዲስ_ኪዳን	ማርቆስ
73	Hyper/hypo	አዲስ_ኪዳን	ጢሞቲዎስ
74	Hyper/hypo	አዲስ_ኪዳን	ተሰሎንቄ
75	Hyper/hypo	ክርስትና_ሀይማኖት	አርቶዶክስ
76	Hyper/hypo	ክርስትና_ሀይማኖት	ክርስቶስ
77	Hyper/hypo	ክርስትና_ሀይማኖት	ኢየሱስ
78	Hyper/hypo	ክርስትና_ሀይማኖት	ካቶሊክ
79	Hyper/hypo	ክርስትና_ሀይማኖት	ፕሮቴስታንት
80	Hyper/hypo	አገልጋይ	ጳጳስ

81	Hyper/hypo	አገልጋይ	መሪጌታ
82	Hyper/hypo	አገልጋይ	ቄስ
83	Hyper/hypo	አገልጋይ	ሰባኪ
84	Hyper/hypo	አገልጋይ	ፓስተር
85	Hyper/hypo	እጽዋት	አትክልት
86	Hyper/hypo	እጽዋት	ፍራፍሬ
87	Hyper/hypo	እጽዋት	የበረሀ_ዛፍ
88	Hyper/hypo	እጽዋት	ሀባብ
89	Hyper/hypo	እጽዋት	አፕል
90	Hyper/hypo	ህመም	ኮሮና
91	Hyper/hypo	ህመም	በሽታ
92	Hyper/hypo	ህመም	ገዳይ
93	Hyper/hypo	ህመም	ትኩሳት
94	Hyper/hypo	ህመም	ወረርሽኝ
95	Hyper/hypo	አሎምፒክ	ሩጫ
96	Hyper/hypo	አሎምፒክ	ኬኒያ
97	Hyper/hypo	አሎምፒክ	ኢትዮጵያ
98	Hyper/hypo	አሎምፒክ	ወርቅ
99	Hyper/hypo	አሎምፒክ	ሜዳሊያ
100	Hyper/hypo	አሰቦት	ቁልቢ
101	Hyper/hypo	አሰቦት	ገዳም
102	Hyper/hypo	አሰቦት	ጻድቃኔ
103	Hyper/hypo	አሰቦት	ዋልድባ
104	Hyper/hypo	አሰቦት	ሸረት

Annex B Human semantic relation score out of 10 for the given word pair and relation type

NO	Similarity_ type	Word pair 1	Word pair 2	WVM Result	expert1	expert2	expert3	expert4	expert5	expert6	expert7	expert8	expert9	expert10	Human mean Semantic relation score
1	Synonym	ጥምቀት	መስቀል	0.99	6	6	8	6	8	6	6	6	8	6	0.75096
2	Synonym	ጥምቀት	በአል	0.99	8	8	8	8	8	8	8	8	8	8	0.75184
3	Synonym	ጥምቀት	ልደታ	0.99	6	8	8	6	8	6	6	6	8	6	0.75137
4	Synonym	ጥምቀት	ገና	0.99	6	8	8	6	8	8	6	8	8	6	0.75208
5	Synonym	ጥምቀት	ኢድ	0.99	6	7	6	6	6	7	6	6	6	6	0.7524
6	Synonym	ፊልም	ሲኒማ	0.99	6	8	8	6	8	6	6	8	8	6	0.75374
7	Synonym	ፊልም	ሚታይ	0.99	8	8	8	8	8	8	8	8	8	8	0.75429
8	Synonym	ፊልም	ተዋናኝ	0.99	6	6	8	6	8	6	6	8	8	6	0.75381
9	Synonym	ፊልም	ምስል	0.99	8	6	8	8	8	6	6	8	8	8	0.75458
10	Synonym	ፊልም	አክተር	0.99	7	8	8	6	6	6	8	8	8	8	0.75474
11	Synonym	ሞባይል	ወሬ	0.99	6	8	8	6	8	8	6	8	8	8	0.755
12	Synonym	ሞባይል	ደወለ	0.99	6	8	8	6	6	6	8	6	6	6	0.75516
13	Synonym	ሞባይል	መተግበሪያ	0.99	8	8	8	8	8	8	8	8	8	8	0.7562
14	Synonym	ሞባይል	ቴክኖ	0.99	8	8	8	8	6	8	8	8	8	8	0.75571
15	Synonym	ሞባይል	መልእክት	0.99	7	8	8	7	8	6	8	8	8	8	0.75544
16	Synonym	ታብሌት	ፓድ	0.99	8	8	8	8	8	8	8	8	8	8	0.75539
17	Synonym	ታብሌት	ኤችፒ	0.99	8	8	6	6	8	6	6	8	6	6	0.75489
18	Synonym	ታብሌት	ዲስክ	0.99	7	8	8	6	7	8	8	8	8	8	0.75575
19	Synonym	ታብሌት	ራም	0.99	6	6	6	8	7	8	6	8	8	8	0.7557
20	Synonym	ታብሌት	ማሸን	0.99	8	8	8	8	8	8	8	8	8	8	0.75624
21	Synonym	ዊነዶው	ባንክ	0.99	7	6	6	6	6	6	6	6	6	6	0.75571
22	Synonym	ዊነዶው	አፊስ	0.99	6	8	8	8	8	8	8	8	8	8	0.75747
23	Synonym	ዊነዶው	ኢንሹራንስ	0.99	6	7	6	7	7	6	6	6	7	6	0.7572
24	Synonym	ዊነዶው	ኡቡንቱ	0.99	8	8	8	8	8	8	8	8	8	8	0.75864

25	Synonym	ዊነዶው	ማክ	0.99	6	8	6	6	8	6	8	6	6	8	0.75813
26	Synonym	ኮምፒውተር	ላፕቶፕ	0.99	8	8	8	8	8	8	8	8	8	8	0.75911
27	Synonym	ኮምፒውተር	ቶሽባ	0.99	8	8	8	8	6	8	6	8	8	8	0.75859
28	Synonym	ኮምፒውተር	ማሽን	0.99	8	8	8	8	8	8	8	8	8	8	0.75857
29	Synonym	ኮምፒውተር	ኤቸፒ	0.99	8	8	8	8	6	8	6	8	8	8	0.75803
30	Synonym	ኮምፒውተር	ፓድ	0.99	8	8	8	8	8	8	8	8	8	8	0.758
31	Synonym	ዘፈን	ሰማ	0.99	6	8	6	6	6	8	6	8	8	6	0.75743
32	Synonym	ዘፈን	አደመጠ	0.99	6	8	6	8	6	8	8	8	8	6	0.75849
33	Synonym	ዘፈን	ቱዲ	0.99	7	8	8	8	8	6	8	8	8	6	0.75903
34	Synonym	ዘፈን	ጥላሁን	0.99	7	8	8	8	8	6	8	8	8	6	0.75915
35	Synonym	ዘፈን	ዘፋኝ	0.99	8	8	8	8	8	8	8	8	8	8	0.75929
36	Synonym	ደራሲ	ተራኪ	0.99	6	8	8	8	8	8	8	8	8	8	0.7587
37	Synonym	ደራሲ	አዘጋጅ	0.99	6	8	8	8	6	6	8	8	6	8	0.75838
38	Synonym	ደራሲ	ድምጻዊ	0.99	7	7	7	6	6	6	6	6	7	6	0.75896
39	Synonym	ደራሲ	አርቲስት	0.99	8	8	6	8	8	8	8	8	8	8	0.76076
40	Synonym	ደራሲ	ኤዲተር	0.99	7	8	8	6	6	6	6	6	6	8	0.76046
41	Synonym	ማሲነቆ	ዋሽንት	0.99	6	8	8	8	8	8	6	8	8	8	0.76188
42	Synonym	ማሲነቆ	በገና	0.99	6	8	8	8	8	8	6	8	8	8	0.7619
43	Synonym	ማሲነቆ	ክራር	0.99	6	8	8	8	8	8	6	8	8	8	0.76194
44	Synonym	ማሲነቆ	ከበሮ	0.99	6	8	8	8	8	8	6	8	8	8	0.76197
45	Synonym	ማሲነቆ	መሳሪያ	0.99	8	8	8	8	8	8	8	8	8	8	0.762
46	Antonym	ትንሽ	በቂ	0.99	8	8	8	8	8	8	8	8	8	8	0.76136
47	Antonym	ትንሽ	ብዙ	0.99	8	8	8	8	8	8	8	8	8	8	0.76069
48	Antonym	ትንሽ	ሆስፒታል	0.96	7	7	7	7	7	7	7	7	7	7	0.76
49	Antonym	ብዛት	ትውልድ	0.99	7	7	7	6	6	6	7	6	6	6	0.76107
50	Antonym	ብዛት	እጥረት	0.99	8	8	8	8	8	8	8	8	8	8	0.76327
51	Antonym	ውድቀት	ግብ	0.99	8	8	8	8	8	8	8	8	8	8	0.76259
52	Antonym	ውድቀት	ተሳካ	0.99	8	8	8	8	8	8	8	8	8	8	0.76189

53	Antonym	ውድቀት	ስኬት	0.99	8	8	8	8	8	8	8	8	8	8	0.76115
54	Antonym	ትርፍ	ባዶ	0.99	8	8	8	8	8	8	8	8	8	8	0.76039
55	Antonym	ትርፍ	ኪሳራ	0.99	8	8	8	8	8	8	8	8	8	8	0.7596
56	Antonym	ለያየ	ጨመረ	0.99	8	8	8	8	8	8	8	8	8	8	0.75878
57	Antonym	ለያየ	መቀላቀል	0.99	8	8	8	8	8	8	8	8	8	8	0.75792
58	Antonym	ለያየ	ማድረግ	0.99	8	8	8	8	8	8	8	8	8	8	0.75702
59	Antonym	መጨምር	አነሳ	0.99	8	8	8	8	8	8	8	8	8	8	0.75609
60	Antonym	መጨምር	ማንሳት	0.99	8	8	8	8	8	8	8	8	8	8	0.75511
61	Antonym	መጨምር	መቀነስ	0.99	8	8	8	8	8	8	8	8	8	8	0.75409
62	Antonym	ጾመ	በላ	0.99	8	8	8	8	8	8	8	8	8	8	0.75302
63	Antonym	ጾመ	ማኘክ	0.99	6	8	8	6	8	8	6	8	8	8	0.7519
64	Antonym	ጾመ	ምግብ	0.99	8	8	8	8	8	8	6	8	8	8	0.7522
65	Antonym	ውሸት	እውነት	0.99	8	8	8	8	8	8	8	8	8	8	0.7515
66	Antonym	ውሸት	ትክክል	0.99	8	8	8	8	8	8	8	8	8	8	0.75026
67	Antonym	ውሸት	ፍጹም	0.99	8	8	8	6	8	8	8	8	8	8	0.74895
68	Antonym	ሩቅ	ከታች	0.99	8	8	8	8	8	8	8	8	8	8	0.74811
69	Antonym	ሩቅ	ቅርብ	0.99	8	8	8	8	8	8	8	8	8	8	0.74667
70	Hyper/hypo	አዲስ_ኪዳን	ፊሊጵስዩስ	0.99	8	8	8	8	8	8	8	8	8	8	0.74514
71	Hyper/hypo	አዲስ_ኪዳን	እብራውያን	0.99	8	8	8	8	8	8	8	8	8	8	0.74353
72	Hyper/hypo	አዲስ_ኪዳን	ማርቆስ	0.99	8	8	8	8	8	8	8	8	8	8	0.74182
73	Hyper/hypo	አዲስ_ኪዳን	ጢሞቲዎስ	0.99	8	8	8	8	8	8	8	8	8	8	0.74
74	Hyper/hypo	አዲስ_ኪዳን	ተሰሎንቄ	0.99	8	8	8	8	8	8	8	8	8	8	0.73806
75	Hyper/hypo	ክርስትና_ሀይማኖት	ኦርቶዶክስ	0.99	8	8	8	8	8	8	8	8	8	8	0.736
76	Hyper/hypo	ክርስትና_ሀይማኖት	ክርስቶስ	0.99	8	8	8	8	8	8	8	8	8	8	0.73379
77	Hyper/hypo	ክርስትና_ሀይማኖት	ኢየሱስ	0.99	8	8	8	8	8	8	8	8	8	8	0.73143
78	Hyper/hypo	ክርስትና_ሀይማኖት	ካቶሊክ	0.99	8	8	8	8	8	8	8	8	8	8	0.72889
79	Hyper/hypo	ክርስትና_ሀይማኖት	ፕሮቴስታንት	0.99	8	8	8	8	8	8	8	8	8	8	0.72615
80	Hyper/hypo	አገልጋይ	ጳጳስ	0.99	8	8	6	8	8	8	6	8	8	8	0.7232

81	Hyper/hypo	አገልጋይ	መሪጌታ	0.99	8	8	8	8	8	8	8	8	8	8	0.72167
82	Hyper/hypo	አገልጋይ	ቄስ	0.99	8	8	6	8	8	8	6	8	8	8	0.71826
83	Hyper/hypo	አገልጋይ	ሰባኪ	0.99	8	8	8	8	8	8	8	8	8	8	0.71636
84	Hyper/hypo	አገልጋይ	ፓስተር	0.99	8	8	6	8	6	8	7	8	8	8	0.71238
85	Hyper/hypo	እጽዋት	አትክልት	0.99	8	8	8	8	8	8	7	8	8	8	0.7105
86	Hyper/hypo	እጽዋት	ፍራፍሬ	0.99	8	7	9	8	8	9	7	8	8	8	0.70632
87	Hyper/hypo	እጽዋት	የበረሀ_ዛፍ	0.99	8	7	9	8	8	8	7	8	8	8	0.70111
88	Hyper/hypo	እጽዋት	ሀባብ	0.99	8	7	8	8	8	8	8	8	8	8	0.69588
89	Hyper/hypo	እጽዋት	አፕል	0.99	8	7	8	8	8	8	8	8	8	8	0.69
90	Hyper/hypo	ህመም	ኮሮና	0.99	7	7	8	8	7	8	8	8	8	8	0.68333
91	Hyper/hypo	ህመም	በሽታ	0.99	8	7	8	8	7	8	8	8	8	8	0.67714
92	Hyper/hypo	ህመም	ገዳይ	0.99	6	6	6	6	7	8	6	8	6	6	0.66923
93	Hyper/hypo	ህመም	ትኩሳት	0.99	8	8	8	8	7	8	8	8	8	8	0.67083
94	Hyper/hypo	ህመም	ወረርሽኝ	0.99	8	8	8	8	8	8	8	8	8	8	0.66
95	Hyper/hypo	አሎምጢክ	ሩጫ	0.99	8	8	8	8	8	8	8	8	8	8	0.646
96	Hyper/hypo	አሎምጢክ	ኬኒያ	0.99	6	6	6	5	6	6	7	7	6	7	0.62889
97	Hyper/hypo	አሎምጢክ	ኢትዮጵያ	0.99	6	6	6	5	6	6	7	6	6	7	0.63
98	Hyper/hypo	አሎምጢክ	ወርቅ	0.99	6	7	6	5	6	7	6	6	6	6	0.63286
99	Hyper/hypo	አሎምጢክ	ሜዳሊያ	0.99	6	6	6	6	8	7	6	6	8	8	0.63667
100	Hyper/hypo	አሰቦት	ቁልቢ	0.99	7	7	6	7	6	6	6	7	5	6	0.63
101	Hyper/hypo	አሰቦት	ገዳም	0.99	7	7	6	7	6	6	6	7	5	6	0.63
102	Hyper/hypo	አሰቦት	ጻድቃኔ	0.99	7	7	6	7	6	6	6	7	5	6	0.63
103	Hyper/hypo	አሰቦት	ዋልድባ	0.99	7	7	6	7	6	6	6	7	5	6	0.63
104	Hyper/hypo	አሰቦት	ሸረት	0.99	7	7	6	7	6	6	6	7	5	6	0.63

Annex C: Questionnaire form offered to Language Experts to List at most 5 similar words for a given query word in a specified relation type.

We kindly ask you to assist us in a psycholinguistic experiment, please list at most top 5 related words for a specified relation type for 20 query words of Amharic language. The aim is measure how much the model retuned answer and much with the experts listed out answer.

Below there is a list words with relation type. For each word, please list at least 1 at most five word for a given relation type. The relation type *Hyper/hypo* stands for Hypernym/hyponym relation between pair of word.

Specific instructions:

- The questionnaire starts on the next page.
- Please fill in your full name at the beginning of the questionnaire.
- Please fill in the scores in the appropriate column of the table.
- Please DO NOT consults your friends on assigning the relatedness scores – it is highly important that the scores you assign be independent of someone else's assessment.

Full Name _____

Relation Type	Word	Answer 1	Answer 2	Answer3	Answer 4	Answer 5
Antonym	ውሸት					
Antonym	ትንሽ					
Antonym	ብዛት					
Antonym	ውድቀት					
Antonym	ትርፍ					
Antonym	ለያየ					
Antonym	መጨምር					
Antonym	ጸመ					
Hyper/hypo	አዲስ_ኪዳን					
Hyper/hypo	ክርስትና_ሀይማኖት					
Hyper/hypo	አገልጋይ					
Hyper/hypo	እጽዋት					
Hyper/hypo	ህመም					
Synonym	ዘፈን					
Synonym	ጥርጣሬ					
Synonym	መጥፎ					
Synonym	ማብራሪያ					
Synonym	አስነመጂ					
Synonym	ሸሸ					
Synonym	ሹክሹክታ					

Annex D Top 5 synset given by three experts

Relation Type	Word	Amharic Word Net Model Returns					Expert1					Expert2					Expert3				
anto	ውሸት	እውነት	ትክክል	ፍጹም			ትክክል	ሀቅ	ትክክል			እውነት	ትክክል				እውነት	ሀቅ	ማመን		
anto	ትንሽ	በቂ	ብዙ	ሆስፒታል			ብዙ	በቂ	በርካታ			ብዙ	በርካታ				ትልቅ	በቂ			
anto	ብዛት	እጥረት	ትውልድ	መድሀኒት			እጥረት	አናሳ				እጥረት	አናሳ				እጥረት	ጥቂት			
anto	ውድቀት	ግብ	ስኬት	ተሳካ			ስኬት	መነሳት				ስኬት	ግብ				ግብ				
anto	ትርፍ	ባዶ	ኪሳራ	ተወ			ኪሳራ					ኪሳራ					ኪሳራ				
anto	ለያየ	ጨመረ	መቀላቀል	ማድረግ			አጣበቀ	መቀላቀል	አዋሀደ			መቀላቀል	ጨመረ				አገናኘ	ጨመረ			
anto	መጨምር	አነሳ	መቀነስ	ማንሳት			ማነሳት	መውሰድ				መቀነስ	ማንሳት				መቀነስ	ማነስ	ማንሳት		
anto	ጾመ	በላ	ማኘክ	ተመገበ			በላ	ምግብ_ወሰደ	ተመገበ			በላ	ተመገበ				ተመገበ	ገደፈ	በላ		

Hyper /hypo	አዲስ_ኪዳን	ፊሊጵስዩስ	እብራውያን	ማርቆስ	ጢሞቲዎስ	ተሰሎንቄ	**ማርቆስ**	ኤርሚያስ	ሉቃስ	ዳንኤል	መጽሀፍ	ማርቆስ	**ፊሊጵስዩስ**	**ጢሞቲዎስ**	ዳንኤል	ማቴዎስ	ኤርሚያስ	ጢሞቲዎስ	ፊሊጵስዩስ	መጽሀፍ	ማቴዎስ
Hyper /hypo	ክርስትና_ሀይማኖት	አርቶዶክስ	ክርስቶስ	ኢየሱስ	ካቶሊክ	ፕሮቴስታንት	**አርቶዶክስ**	**ካቶሊክ**	**ፕሮቴስታንት**			ፕሮቴስታንት	አርቶዶክስ	ካቶሊክ			ፕሮቴስታንት	ካቶሊክ	አርቶዶክስ		
Hyper /hypo	አገልጋይ	ጳጳስ	መሪጌታ	ቄስ	ሰባኪ	ፓስተር	ቄስ	ፓስተር	ሼህ	ጳጳስ	ሰባኪ	ዘማሪ	ሼህ	ቄስ	ፓስተር	ሰባኪ	ቄስ	ፓስተር	ሼህ	ጳጳስ	መነኩሴ
Hyper /hypo	እጽዋት	አትክልት	ፍራፍሬ	የበረሀ_ዛፍ	ሀባብ	አፕል	ተክል	**አትክልት**	ገራር	**የበረሀ_ዛፍ**	ዛፍ	ተክል	ዛፍ	**ፍራፍሬ**	የበረሀ_ዛፍ	እንሰት	ተክል	አትክልት	ፍራፍሬ	የበረሀ_ዛፍ	ዋርካ
Hyper /hypo	ህመም	ኮሮና	በሽታ	ገዳይ	ትኩሳት	ወረርሽኝ	**ኮሮና**	**በሽታ**	ኩላሊት	**ወረርሽኝ**	ካንሰር	ራስምታት	በሽታ	ወባ	ወረርሽኝ	**ትኩሳት**	ኮሮና	በሽታ	ስኳር	ወረርሽኝ	ትኩሳት
WV Syno	ዘፈን	ሙዚቃ	አደመጠ	ቴዲ	ዘፋኝ	ጥላሁን	**ሙዚቃ**	**ዘፋኝ**	ዜማ			ድምጽ	ዜማ				ሙዚቃ	ዘፋኝ	የሚስማ		
WV Syno	ጥርጣሬ	አለማመን	ባዶ	ሰጋት	ፍራቻ	እስር	**አለማመን**	መጠርጠር	እምለት_የለሽ			አለማመን	**ሰጋት**				አለማመን	መጠርጠር	ሰጋት		
WV Syno	መጥፎ	ክፋ	ዘጊ	ሀዘን	ጉዳት	ጎጂ	**ክፋ**	አስቀያሚ	**ጎጂ**			አሳዛኝ	አስቀያሚ	ጎጂ			ክፋ	አስቀያሚ	ቀፋፊ		

WV Syno	ማብራሪያ	ገለጻ	አቀረበ	ሰጠ	ተናገረ	ጻፈ	**ገለጻ**	መግለጫ				ገለጻ	**ትንታኔ**				ትንታኔ	ማሰረዳት			
WV Syno	አስጎመጀ	ተወዳጅ	ማራኪ	ቆንጆ	ሚያምር	ውብ	ሚያሳሳ	**ማራኪ**	**ዉብ**			ሳቢ	አስገራሚ	ዉብ			ሳቢ	ማራኪ	ዉብ		
WV Syno	ሸሸ	አመለጠ	ሮጠ	ተቀመጠ	ዞረ	ተደበቀ	**ተደበቀ**	**አመለጠ**				ተደበቀ					ጠፋ	አመለጠ			
WVSyno	ሹክሹክታ	ጭምጭምታ	ወሬ	ፕሮፖጋንዳ	ተነዛ	ድብቅ	**ጭምጭምታ**	የስሚስሚ	በጆሮ_መንገር			ጭምጭምታ	**ሀሜት**				ያልተረጋገጠ	ሀሜት			
DS Syno	ዘፈን	ሙዚቃ					**ሙዚቃ**	**ዘፋኝ**	ዜማ			ድምጽ	ዜማ				ሙዚቃ	ዘፋኝ	የሚስማ		
DS Syno	ጥርጣሬ	አለማመን					**አለማመን**	መጠርጠር	እምለት_የለሽ			አለማመን	**ሰጋት**				አለማመን	መጠርጠር	ሰጋት		
DS Syno	መጥፎ	አስቀያሚ					**ክፋ**	አስቀያሚ	**ጎጂ**			አሳዛኝ	አስቀያሚ	ጎጂ			ክፋ	አስቀያሚ	ቀፋፊ		
DS Syno	ማብራሪያ	ትንታኔ					**ገለጻ**	መግለጫ				ገለጻ	**ትንታኔ**				ትንታኔ	ማሰረዳት			
DS Syno	አስጎመጀ	ሳቢ					ሚያሳሳ	**ማራኪ**	**ዉብ**			ሳቢ	አስገራሚ	ዉብ			ሳቢ	ማራኪ	ዉብ		
DS Syno	ሸሸ	ተደበቀ					**ተደበቀ**	**አመለጠ**				ተደበቀ					ጠፋ	አመለጠ			

DS Syno	ሹክሹክታ	ሀሜት	ወሬ				ጭም ጭምታ	የስ ሚስ ሚ	በጆሮ_ መንገር			ጭም ጭምታ	ሀሜት				ያልተ ረጋገ ጠ	ሀሜት			

Signed Declaration Sheet

I, the undersigned, declare that this research is my original work and has not been presented for degree in any other university, and that all sources of materials used for the research have been acknowledged.

Declared by:

Name: Mulat Getaneh

Signature: _____

Date: _____

Confirmed by advisor:

Name: Yaregal Assabie (PhD)

Signature: _____

Date: _____