Dictionary Based Word Sense Disambiguation for Amharic

Gashaw Sisay
HiLCoE, Computer Science Programme, Ethiopia
gsisay@gmail.com

Solomon Teferra
School of Information Science, Addis Abeba University, Ethiopia
solomon_teferra_7@yahoo.com

Abstract

A word carrying multiple senses is one source of ambiguity in natural languages that people have to deal with by asking for clarification, using their real world experience or the context as a means to deduce the intended sense of the ambiguous word in that specific communication.

However, computers lack the capacity to identify the intended sense of an ambiguous word that humans have. This paper tries to build a system that can do Word Sense Disambiguation (WSD) on a given sentence by identifying the ambiguous word and using Machine Readable Dictionary (MRD) to infer the most likely meaning of the ambiguous word and conduct performance comparison of the MRD based Lesk Algorithm variants to identify which algorithm is more effective.

The result of the evaluation showed that Simplified Lesk algorithm with stemming has the highest precision score of 0.418 followed by the Original Lesk algorithm with stemming achiving a 0.291 score of precision. Both the Simplified Lesk and Original Lesk algorithm versions without stemming scored a very low precision score of 0.018.

Keywords: Word Sense Disambiguation; Machine Readable Dictionary; Lesk Algorithm Variants; Knowledge Based Approach

1. Introduction

It is a fact that words in natural language have multiple senses (meanings) that can be identified based on context or real world experience. This polysemic (a word having two or more different but related meanings) and homonymic property of words is one source of ambiguity in communication. Amharic also has this feature of natural languages that is a challenge for computers to deal with in processing natural languages. For example, the word '*ሰራተኛ*' could mean 'a person engaged in a line of work' or 'a person diligent at his/her work'. Both of these meanings of the word '*ሰራተኛ*' are related with the word '*ሰራ*' ('work'). Two or more words are considered homonyms if they are either homophones (sound the same) or homographs (have same spelling) or if they are both homophones and homographs but do not have related meanings. For example, the words '*ሠማ*' ('to become hot or warm') and '*ሰማ*' ('to listen') are homonyms that are homophones but not homographs, the geminating word '*ገና*' ('christmass') and '*ገና*' ('yet to be') are also homonyms that are homographs but not homophones, the word '*ሳለ*' (meanings 'to make a painting' or 'to sharpen' or 'to cough') also show the property of homonymy in that it is homophone and homograph but with completely different meanings [12].

When people face ambiguity in word meanings, they do the disambiguation task from their real world experiences or by asking for clarification, in the case of conversations, which computer systems are incapable of doing. This problem of WSD is an ongoing research area in Natural Language Processing (NLP) where there is a need to come up with better techniques that enable computer systems do disambiguation tasks by themselves.

WSD is the automated activity of finding the meaning of words in a given context when ambiguities arise [1]. NLP activities that require identifying the exact meaning of ambiguous words like Machine Translation (MT), Speech Recognition (SR), Information Retrieval (IR), and Information Extraction (IE) [2] need to incorporate a Word Sense Disambiguation component to increase their accuracy or effectiveness.

One can find NLP research works on Ethiopian languages that cover most of the research areas of NLP even though there is still a lot left to be explored considering the number of languages Ethiopia have and the available features in each language. The research works in [4] (using WordNet), in [5, 9] (using Semantic Similarity) are based on knowledge-based approach, in [3] (Semi-Supervised learning) and [8] are based on machine learning approach for the Amharic WSD problem.

Machine Readable Dictionary (MRD) based approach is a Knowledge Based approach where the disambiguation process is done based on some kind of written document such as human language dictionary, WordNet or thesaurus. This approach uses a list of lexicons and their corresponding meanings in a dictionary to do the WSD [6]. For Amharic WSD problem this approach is unexplored that needs investigation.

The works in [3, 8] that are based on the machine learning approach have performance limited to the ambiguous words they are trained on and preparing a corpus at least for the majority of the ambiguous words is very challenging. The researches that follow the knowledge based approach donot have the problem of preparing training data but they face their own problem of not having an adequate knowledge source and in the worst case none at all. Amharic has a dictionary knowledge source, where WordNet or Thesaurus is not available for NLP applications. Due to this, the work in [4] conducted for Amharic WSD based on WordNet has to construct Amharic WordNet limited to 10,000 synsets and 2,000 words from a dictionary.

Creating a practical solution for the Amharic WSD problem based on these researches has its own challenges. It is impractical to prepare a training data for all ambiguous words which makes machine learning researches unsuitable and the WordNet based Amharic WSD has its own limitation of word size (having only 2,000 words). On the other hand, using dictionary as a knowledge source, one can create a practical solution that only requires having a database of Amharic-to-Amharic dictionary. This paper aims at creating a solution for the Amharic WSD problem using Amharic-to-Amharic dictionary of more than 25,000 words [7] to infer the most likely meaning of the ambiguous word and provide performance comparison of MRD based Lesk Algorithm variants.

2. Related Work

The work in [3] was conducted to develop Word Sense Disambiguation Model for Amharic Words using Semi-Supervised Learning approach. The experiment was conducted using five selected ambiguous Amharic words with a total of 1031 datasets, two clustering algorithms (simple K-means and EM) and five classification algorithms (AdaboostM1, bagging, ADtree, SMO and Naïve Baye). The experiment result shows performance score of 88.47% using ADtree, 87.40% using SMO and 83.94% using AdaboostM1 and window size of 2-2 and 3-3 can be an optimal window size for Amharic WSD systems development using Semi-Supervised Learning approach.

The research work of Solomon Mekonnen [8] is machine learning based on the Amharic WSD problem by applying a supervised machine learning approach to a corpus of Amharic sentences. In this study, a total of 1045 English sense examples for the five ambiguous words such as አጠና (eTena), መሳል (mesal), መሣሣት (me`sa`sat), መጥራት (metrat), and ቀረጸ (qereSe,) were used. Datasets were collected from British National Corpus (BNC) and the sense examples were translated to Amharic using dictionary and preprocessed to make them ready for the experiment. Then, Naive Bayes classifier was employed from Weka 3.62 package to perform the supervised learning on the preprocessed dataset using 10-fold cross-validation. The author evaluated the classifiers for the five ambiguous words and achieved accuracy within the range of 70% to 83% which was very encouraging and concluded that window size of 3-3 on both side of the ambiguous words is enough to disambiguate.

The effectiveness of semantic similarity measure in WSD for query expansion was investigated in [9]. A simple Word-Net (words with their synonym and gloss definitions) was constructed to be used as a knowledge base in identifying the sense of a given query by applying semantic similarity measure.

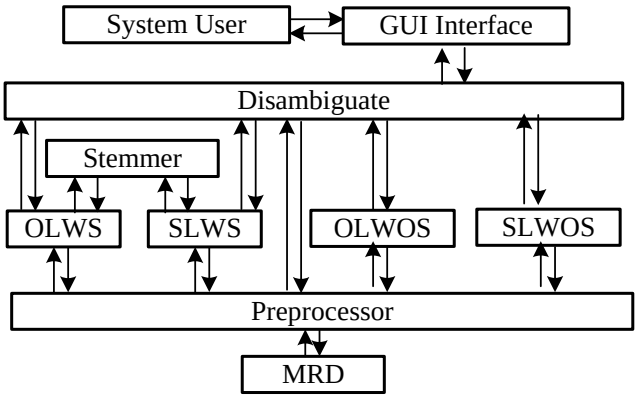
Using the idea of Lesk algorithm, WSD was performed with two methods: gloss to gloss and synset to gloss by comparing information associated with its synonyms and gloss definition with reference to Amharic Word-Net. The combination of the two disambiguation methods formulates the modified query and used for expanding the original query. Finally, the query expansion module is integrated with Information Retrieval system to show the enhancement of Amharic IR system performance. Accordingly, the study has reported that the method using synset for query expansion registered performance of 59% F-measure. Besides, this method registered 6% improvement from the original query.

A WordNet based work by Seid Hassen Yesuf and Yaregal Assabie [4] is the only Knowledge-Based research in the Amharic WSD problem. This research work developed a word sense disambiguation system for Amharic using Amharic WordNet. Due to the nonexistence of WordNet for Amharic, the authors constructed a WordNet containing 10,000 synsets and 2,000 words using Amharic dictionary as a source. The use of WordNet provided senses of words along with the relation these words have with other words to be used in the disambiguation process. Evaluation of the system was made using precision and recall by taking into consideration context window sizes and morphological analyzer effect on 200 randomly selected sentences containing ambiguous words with two or three frequently used senses taken from newspapers and linguistic experts as a test data. The result shows that using this approach an accuracy of 57.5% and 80% can be achieved with and without morphological analyzer, respectively.

The work in [6] is for the Bangla language using a dictionary based approach which is the most related research work to this paper. The research objective was to come up with a system that can disambiguate noun, adjective and verb ambiguous words in Bangla language sentences which have multiple senses by using a dictionary lookup. The research proposed two algorithms where the parsing algorithm creates a parse tree from an input sentence to verify the words existence in the Bangla Language dictionary and the validity of the sentence structure while the detect algorithm looks for ambiguous words in the input Bangla sentences by looking for which word is most overlapped with the dictionary definitions and also its actual meaning. Evaluating the system with limited sentences and without considering semantic features of a sentence, an overall accuracy of 82.40% was achieved.

3. The Proposed Solution

The WSD system uses an Amharic sentence/phrase written in Ethiopic script (fidel) and entered into the system by the user as input. The second input to the system is the dictionary definitions of a word fetched from the MRD when requested by the WSD system. Each of these two inputs go through processes (preprocessing, stemming and algorithmic evaluation) to detect and disambiguate ambiguous words. After this, the most likely dictionary definition of the detected ambiguous word is returned as an output. Figure 1 shows the design of the WSD system which is followed by each module's description.



OLWS - Original Lesk With Stemming OLWOS - Original Lesk With Out Stemming
SLWS - Simplified Lesk With Stemming SLWOS - Simplified Lesk With Out Stemming

Figure 1: Design of the WSD System

A. GUI Module

The user interface has three parts. The first one is where users input a sentence in Ethiopic script to be disambiguated and a button to start the process. The second part shows the result of the disambiguation process in short while the last part shows in detail each step of the disambiguation process. Figure 2 shows the graphical user interface.

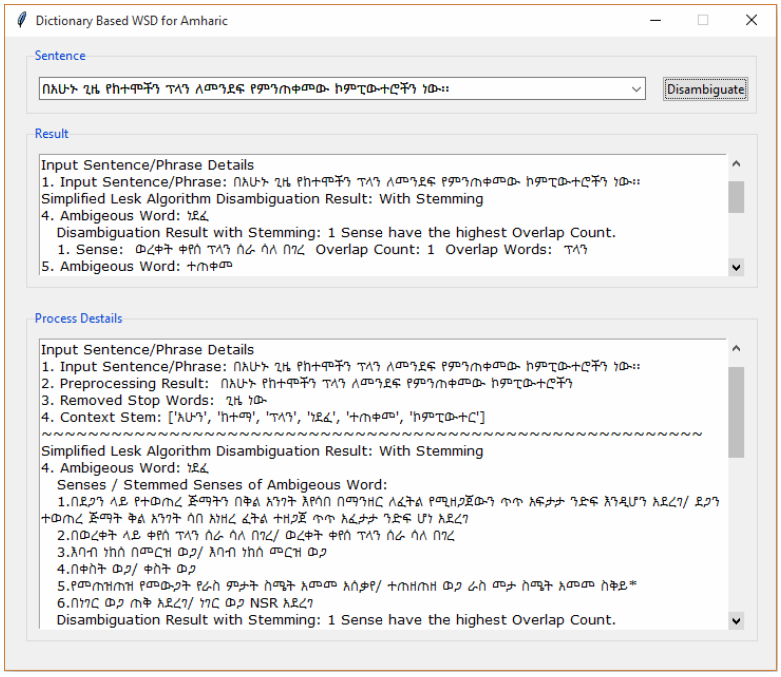


Figure 2: GUI

B. Preprocessing Module

This module removes different components to make the sentence/phrase more suitable for the disambiguation task by breaking down the input sentence/phrase into a list of words (tokenization) and removing items that are not relevant for the disambiguation task like punctuation marks, numbers and other symbols that are not letters. Stop words (words that do not carry meaning) are also removed using a pre-prepared list of stop words as a reference from the MRD. This module is used by the disambiguate module to preprocess user input sentence/phrase while original lesk with stemming, original lesk without stemming, simplified lesk with stemming and simplified lesk without stemming modules use it to preprocess dictionary definitions fetched from the MRD.

C. Disambiguate Module

This module handles the coordination activity of all the other modules of the WSD system.

D. Stemmer Module

This module is constructed by integrating the HornMorpho L3 library (a morphological analysis tool for Amharic, Afan Oromo and Tigrigna languages) with a code to extract the stem/root part of the morphological analysis result. The module stems a single word passed to it as a parameter and it returns the stem of the word. If the word does not have a stem for some reason, the module returns the phrase 'NSR' (No Stemming Result). To stem a word, it is saved in a text file with UTF-8 encoding so that the file content is considered as a Unicode text. L3.anal() method of the HornMorpho library with parameters of language marker 'am', input text file, UTF-8 encoded text file for output, citation enabled, grammar and raw result disabled is run. The UTF-8 encoded output text file is then processed to extract the stem/root of the word among other results of the morphological analysis.

E. MRD Database Module

This module organizes the words with their definitions and a pre-compiled list of stop words using five tables in a single database. *StopWordList* table holds only Amharic stop words compiled from [10, 11] for the purpose of referencing stop words so that the system can detect and remove stop words both from the input sentences and dictionary definitions of words. *WordList* table holds a list of words with their POS and the number of definitions it has compiled from the Amharic-to-Amharic dictionary [7]. *DefinitionList* table contains the definitions of all the words from the *WordList* table with their definition number. *ReferenceList* table holds one word referring to another word where the first word does not have a definition entry in the dictionary while the second word has. *SubWordList* holds list of words that are derived from other words to show which word is the root and which one is the derived word. Figure 3 shows the structure of the MRD Database.

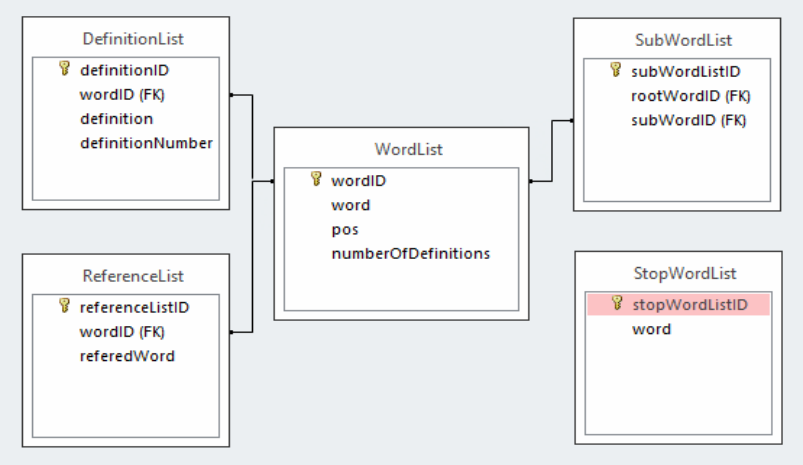


Figure 3: Structure of the MRD Database

F. Original Lesk Algorithm with Stemming (OLWS) Module

After applying preprocessing and stemming on the input sentence/phrase, this module looks for ambiguous words in the input by checking how many definitions each word has in the *WordList* table. A value of two or more number of definitions qualifies a word to be ambiguous. For each ambiguous word detected, dictionary definitions are fetched, preprocessed and stemmed to do count of common words among the definitions of each word in the input and the senses of the ambiguous word. The ambiguous word definition with the highest overlap count is returned to the Disambiguate module. 'Zero or Equal Overlap Count' is returned for zero or equal number of overlap.

G. Original Lesk Algorithm without Stemming (OLWOS) Module

It is similar in function and operation to the previous module except that all the overlap

calculations are done without applying stemming to the words involved in the process.

H. Simplified Lesk Algorithm with Stemming (SLWS) Module

This module preprocesses and stems the input and then looks for ambiguous words in the input sentence/phrase by checking how many definitions each word has in the *WordList* table. A value of two or more number of definitions qualifies a word to be ambiguous. For each ambiguous word detected, dictionary definitions are fetched, preprocessed and stemmed to do count of common words among the definitions of each word in the input and the context. The ambiguous word definition with the highest overlap count is returned to the Disambiguate module. 'Zero or Equal Overlap Count' is returned for zero or equal number of overlap.

I. Simplified Lesk Algorithm without Stemming (SLWOS) Module

This module is similar in function and operation to the previous module except that all the overlap calculations are done without applying stemming to the words involved in the process.

J. Flow of Operation

Disambiguation of the test data follows a sequence of operations coordinated by the Disambiguate module. The list provided below shows the steps taken.

i. User inputs a sentence to be disambiguated and clicks on the Disambiguate button. The Disambiguate module is called and the disambiguation process starts.

ii. Disambiguation module gives control with the input sentence/phrase to Simplified Lesk with stemming module. This module disambiguates the input after applying preprocessing and stemming it and returns the result with control to the Disambiguate module. After this, Disambiguate module passes the result to the GUI module to be displayed.

iii. Disambiguation module gives control with the input sentence/phrase to Simplified Lesk without stemming module. This module disambiguates the input after preprocessing it and returns the result with control to the Disambiguate module. After this, Disambiguate module passes the result to the GUI module to be displayed.

iv. Disambiguation module gives control with the input sentence/phrase to Original Lesk with stemming module. This module disambiguates the input after preprocessing and stemming it and returns the result with control to the Disambiguate module. After this, Disambiguate module passes the result to the GUI module to be displayed.

v. Disambiguation module gives control with the input sentence/phrase to Original Lesk without stemming module. The module disambiguates the input after preprocessing it and returns the result with control to the Disambiguate module. After this, Disambiguate module passes the result to the GUI module to be displayed.

vi. Disambiguation module exits and control is returned to the main loop of the system.

4. Discussion

The performance of the algorithms used in the disambiguation process was measured in terms of precision and recall. Precision was defined as the *number of words correctly disambiguated over total number of words disambiguated* while Recall was defined as the *number of words disambiguated over the total number of words attempted to be disambiguated*, according to the definition of precision and recall measure for WSD proposed in [13].

Test data (a set of sentences for each and every sense of 25 ambiguous words selected from [7] with two or more distinict senses) and MRD (a database of words with their senses acquired from [7] and a list of stopwords compiled from [10, 11]) were given to the WSD system as input and the result was analysed.

MRD based WSD depends on the count value of common words between context (the dictionary definition of the words in the context) and dictionary definitions of the ambiguous word. Since this count value is based on comparing words similarity, words that are derived from the same root word are

considered different words and fail the common word count. To remedy this situation, stemming was applied on every word before any word similarity comparison was made in the algorithms with stemming version. Table 1 shows the stemming result of the test data and definitions/senses from the MRD.

Table 1: Stemming Result

Data Source	*Correct*	*Incorrect*
Test Data	94%	6%
Definitions/Senses	95%	5%

Application of stemming in the Original Lesk Algorithm resulted in 0.27 and 0.84 increase in precision and recall values, respectively. In the Simplified Lesk Algorithm, it has also caused 0.4 and 0.82 increase in precision and recall values, respectively.

Evaluation of the algorithms' performance (Table 2) shows that Simplified Lesk with stemming has the highest precision (0.42) while Original Lesk with stemming produced the second highest precision (0.29). Out of all the disambiguation attempts made, Simplified Lesk with stemming resulted in a small error of 5% whereas 25% error had occurred in the Original Lesk with stemming. The lowest result for precision (0.02) is recorded in both the Original Lesk and Simplified Lesk algorithms without applying stemming.

Table 2: Performance in terms of Precision and Recall

Algorithm	*Precision*	*Recall*
Original Lesk with Stemming	0.29	0.96
Simplified Lesk with Stemming	0.42	0.96
Original Lesk without Stemming	0.02	0.13
Simplified Lesk without Stemming	0.02	0.15

5. Conclusion

This paper presented a system for the Amharic WSD problem originating from Polysemy and Homonymy using Amharic-to-Amharic dictionary as a knowledge source and provided performance comparison of the MRD based Lesk algorithm variants used in the disambiguation process.

In conclusion, stemming has a positive influence in precision and recall by converting words into their root form for detection and disambiguation. However, it has also caused the number of incorrect disambiguation results to increase as more words are disambiguated than before. In both with and without stemmer versions of the algorithms, Simplified Lesk with stemming has the highest performance score and the application of stemming enhanced the algorithms' performance.

References

[1] Roberto Navigli, "Word Sense Disambiguation: A Survey", ACM Computing Surveys, Vol. 41, No. 2, February 2009.

[2] Alok Ranjan Pal and Diganta Saha, "Word Sense Disambiguation: A Survey", International Journal of Control Theory & Computer Modeling (IJCTCM), Vol. 5, No. 3, July 2015.

[3] Getahun Wassie, Ramesh Babu, Solomon Teferra, and Million Meshesha, "A Word Sense Disambiguation Model for Amharic Words using Semi-Supervised Learning Paradigm", STAR Journal, July-Sep 2014: 3(3), pp. 147-155.

[4] Seid Hassen Yesuf and Yaregal Assabie, "Amharic Word Sense Disambiguation Using WordNet", 5th International Conference on the Advancement of Science & Technology (ICAST-2017), Bahir Dar University, Ethiopia, May 2017.

[5] Teshome Kassie, "Word Sense Disambiguation for Amharic Text Retrieval: A Case Study for Legal Documents", Unpublished Masters Thesis, Addis Ababa University, 2009.

[6] A. Haque and M. M. Haque, "Bangla Word Sense Disambiguation System Using Dictionary Based Approach", Proceedings of the International Conference on Advances in Information & Communication Technology, 2016.

[7] አማርኛ መዝገበ ቃላት፤ የኢትዮጵያ ቋንቋዎች ጥናትና ምርምር ማእከል፤ አዲስ አበባ ዩኒቨርሲቲ፤ የካቲት 1993.

[8] Solomon Mekonnen, "Word Sense Disambiguation for Amharic Text: A Machine Learning Approach", Unpublished Masters Thesis, Addis Ababa University, 2010.

[9] Samrawit Zewdneh, "Word Sense Disambiguation using Semantic Similarity for

Query Expansion in Amharic Information Retrieval", Unpublished Masters Thesis, Addis Ababa University, 2014.

[10] Tihitina Petros Degsew, "Modeling and Designing Amharic Query System to Bilingual (English-Amharic) Databases", Unpublished Masters Thesis, Addis Ababa University, 2014.

[11] Mequannint Munye Zeru, "Amharic-English Bilingual Search Engine", Unpublished Masters Thesis, Addis Ababa University, 2010.

[12] PEDIAA, www.pediaa.com/difference-between-polysemy-and-homonymy/, last accessed on April 2019.

[13] Rada Mihalcea and Dan Moldovan, “A Highly Accurate Bootstrapping Algorithm for Word Sense Disambiguation”, International Journal on Artificial Intelligence Tools, World Scientific Publishing Company, Vol. 10, Nos. 1 & 2, 2001.