# Word Sense Disambiguation for Wolaita Language Using Machine Learning Approach


Temesgen Tadesse Tilahun

A Thesis Submitted to

The Department of Computer Science and Engineering

School of Electrical Engineering and Computing

Presented in Partial Fulfillment of the Requirement for the Degree of Master's in Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

August 2021

ASTU, Adama, Ethiopia

# Word Sense Disambiguation for Wolaita Language Using Machine Learning Approach

Temesgen Tadesse Tilahun

Advisor Name: Dr. Mesfin Abebe (Ph.D.)

A Thesis Submitted to

The Department of Computer Science and Engineering

School of Electrical Engineering and Computing

Presented in Partial Fulfillment of the Requirement for the Degree of Master's in Science in Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

August 2021

ASTU, Adama, Ethiopia

# Declaration

I hereby declare that this Master Thesis entitled "**Word Sense Disambiguation for Wolaita Language Using Machine Learning Approach**" is my original work. That is, it has not been submitted for the award of any academic degree, diploma, or certificate in any other university. All sources of materials that are used for this thesis have been duly acknowledged through citation

| _____ | _____ | _____ |
|---|---|---|
| Name of the Student | Signature | Date |

## Recommendation

I/we, the advisor(s) of this thesis, hereby certify that I/we have read the revised version of the thesis entitled "**Word Sense Disambiguation for Wolaita Language Using Machine Learning Approach**" prepared under my/our guidance by **Temesgen Tadesse Tilahun** submitted in partial fulfillment of the requirements for the degree of Mater's of Science in Computer Science Engineering. Therefore, I/we recommend the submission of the revised version of the thesis to the department following the applicable procedures.

| _____ | _____ | _____ |
|---|---|---|
| Advisor | Signature | Date |

## Approval Page of M.Sc. Thesis

I/we, the advisors of the thesis entitled "**Word Sense Disambiguation for Wolaita Language Using Machine Learning Approach**" and developed by **Temesgen Tadesse Tilahun**, hereby certify that the recommendation and suggestions made by the board of examiners are appropriately incorporated into the final version of the thesis.

| _____ | _____ | _____ |
|---|---|---|
| Advisor | Signature | Date |

We, the undersigned, members of the Board of Examiners of the thesis **Temesgen Tadesse Tilahun** have read and evaluated the thesis entitled "**Word Sense Disambiguation for Wolaita Language Using Machine Learning Approach**" and examined the candidate during open defense. This is, therefore, to certify that the thesis is accepted for partial fulfillment of the requirement of the degree of Master of Science in Computer Science Engineering

| _____ | _____ | _____ |
|---|---|---|
| Chairperson | Signature | Date |
| _____ | _____ | _____ |
| Internal Examiner | Signature | Date |
| _____ | _____ | _____ |
| External Examiner | Signature | Date |

Finally, approval and acceptance of the thesis is contingent upon submission of its final copy to the Office of Postgraduate Studies (OPGS) through the Department Graduate Council (DGC) and School Graduate Committee (SGC).

| _____ | _____ | _____ |
|---|---|---|
| Department Head | Signature | Date |
| _____ | _____ | _____ |
| School Dean | Signature | Date |
| _____ | _____ | _____ |
| Office of Postgraduate Studies, Dean | Signature | Date |

# ACKNOWLEDGMENTS

Firstly, I would like to acknowledge my Lord Jesus, who helped in all aspects, to conduct my research. Next to God, I would like to acknowledge my advisor, Dr. Mesfin Abebe for his advice, his critical comments, and his commitment to guide this thesis. Not only that, but also he advised me not only to conduct this thesis, but also to be one of research scholar and to acquire the attitudes of research scholars

I would also like to acknowledge the members of the AI SIG of the CSE department. Including the SIG supervisor Dr. Ahmed Waze, for his continued commitment to guide and support this research from its inception to completion. And to all members who sat on each phase presentation panel for their inputs which greatly shaped and improved the work. I would like to thank all my classmates who are working on their master's thesis for their timely help, idea, and support until the completion of my thesis.

My special thanks also to Wolaita Language and Literature Department staffs for their commitment to annotate and provide quality data for our study. I also give special thanks to Wolaita Wogeta FM staffs specially Mr. Paulos Tekile for his contribution to convert hand-written news to electronic form and to translate Amharic new into Wolaita language.

Last but not least, my thanks go to my family and my spouse Mrs. Buzayehu Belete for their every support and advice in my life. May God always Bless and keep them safe.

# TABLE OF CONTENTS

# LIST OF TABLES

**TABLES** **PAGES**

# LIST OF FIGURES

**FIGURE** **PAGE**

# LIST OF PSEUDO CODE

**PSEUDO CODE** **PAGE**

# ABBREVIATIONS AND ACRONYMS

| | |
|---|---|
| **AI** | Artificial Intelligence |
| **BNC** | British National Corpus |
| **BOW** | Bag of Words |
| **CBOW** | Continuous Bag of Words |
| **CV** | Cross-Validation |
| **EM** | Expectation Maximization |
| **FBC** | Fana Broadcasting Corporation |
| **FM** | Frequency Modulation |
| **IDF** | Inverse Document Frequency |
| **IE** | Information Extraction |
| **IR** | Information Retrieval |
| **K-NN** | K-Nearest Neighbors |
| **MRD** | Machine-Readable Dictionary |
| **MT** | Machine Translation |
| **NLP** | Natural Language Processing |
| **NLTK** | Natural Language Toolkit |
| **NLU** | Natural Language Understanding |
| **POS** | Parts of Speech |
| **RFC** | Random Forest Classifier |
| **SGD** | Stochastic Gradient Descent |
| **SNNP** | South Nation Nationality and People |
| **SPSS** | Statistical Package for Social Science |
| **SVC** | Support Vector Classifier |
| **SVD** | Singular Value Decomposition |
| **SVM** | Support Vector Machine |
| **TF** | Term Frequency |
| **TV** | Television |
| **UML** | Unified Modeling Language |

| | |
|---|---|
| **URL** | Uniform Resource Locator |
| **WED** | Wolaita English Dictionary |
| **WS** | Window Size |
| **WSD** | Word Sense Disambiguation |
| **WSGI** | Web Server Gateway Interface |
| **WSJ** | Wall Street Journal |

# ABSTRACT

*The amount of data accessible online has been increasing and the need for Natural Language Processing significantly increasing to access and process this data. However, ambiguity problems have faced the difficulties for Natural Language Processing. As human beings, computers can't understand one word in different way. As solution to this, Word Sense Disambiguation models developed for many languages to address the problem of lexical ambiguity. For the Wolaita language, there are also a lot of polysemy words and these can be the cause of difficulties for Natural Language processing applications developed by previous researchers. Therefore, Word Sense Disambiguation Model for Wolaita language using a machine learning approach was proposed. To conduct the research, a total of 2797 sense examples were collected from Holy Bible, academic books, media agencies (Sport, Health, Business and national and international News), and data from prior researchers. The collected data was annotated by the language experts and then five datasets prepared for five ambiguous words such as "Doona", "Ayfiya", "Aadhdha", "Naaga" and "Ogiya". We employed quantitative experimental research approach to determine the best combination of the machine learning algorithms and features extraction techniques. Support Vector Classifier , Bagging, Random Forest Classifier, and AdaBoost classifier with BOW, TF-IDF and Wor2Vec feature extraction techniques selected and trained using five datasets on six-window sizes (WS3, WS5, WS7, WS9, WS11 and WS13). From the six window sizes, WS11 (5-5) was selected as the optimal window size in terms of accuracy and the computational time it costs. Among four algorithms, Support Vector Classifier and Bagging classifiers with TF-IDF achieved accuracy of* 83.22*% and 82.82 % respectively on WS11 (5-5) using 10-fold cross-validation.*

***Key Words**: Word Sense Disambiguation, Natural Language Processing, Window Sizes, Machine Learning Algorithms, Feature Extraction, Wolaita language*

# CHAPTER ONE

# 1. INTRODUCTION

## 1.1 Background of the Study

Nowadays, a large amount of natural language data have been processed by NLP, and It has wider applications including MT, speech recognition, IR and Web-based question answering. The goal of NLP is to improve human-human communication, human-computer communication by enabling useful processing of text and speech (Daniel Jurafsky & James H. Martin, 2006). However, the most challenging issue in NLP is ambiguous words by which computers can't identify the sense of each word in the context of the given text or Speech (Choueka, 2014). But, human beings can easily understand the sense of a word in text or speech. As solution to for ambiguity challenges, WSD was developed and integrated with NLP applications. As defined by Scholarpedia (Philip & Eneko , 2008), WSD is the process of identifying the meaning of the ambiguous word in a particular context by activating the word. Therefore, many researchers conducted WSD for other languages. In our study, we conducted WSD for Wolaita language.

Wolaita is one of the North Omotic language spoken groups in the Wolaita Zone and some other parts of the SNNP Region of Ethiopia (WAKASA, 2014). The language is the official language in the Wolaita Zone of Ethiopia. According to statistics of Ethnology, 5% to 25% of the population are literate in this language. Now the language speakers in Ethiopia are above 5,908,000 million (Toophphiya Hayo Bilaa Woshettaa, 2007).

Wolaita has existed in written form since the 1940s when the Sudan Interior Mission first devised a system for writing it by Latin script. Later the team led by Dr. Bruce Adams revised the writing system of the language. The sentences in the language formed by SOV agreement and the lexical stem has the only suffix but not prefix. The gemination of vowels and consonants in the particular word and the elongation of suffix make the morphology complex for Wolaita language (Girma, 2018). As other languages, there are polysemous words in the wolaita language and these can make different kinds of ambiguities. Among these ambiguity the most challenging issues for NLP is semantic lexical ambiguity that occurs at the word level and the

word or lexical items may have more than one meaning according to the context (NG'ANG'A, 2005). For example, the word '**doona'** has more than one meaning according to the context of the sentences described below.

- ✓ 'Taani Wolayttatto **doonan** haasayiyoogaa danddayays.(' I can speak wolaita **language'**)
- ✓ 'Tani ta **doonan** eessaa mays. ('I eat honey by my **mouth**.)
- ✓ 'Nuuni abbaa **doonan** de'iya allaana be'ida'(we saw the city located **near** the lake)

The meaning for '**doona'** in the first sentence is '**the language**' and in the second sentence is '**the mouth**' and in the third sentence is **'near**' in English.

WSD was developed for many languages using knowledge-based approaches, corpus-based approaches, and hybrid approaches which is the combination of knowledge-based and corpus-based approaches. In knowledge-based approaches, disambiguation was done by finding pair of dictionary senses with the highest word overlap in their definition in the dictionary using WordNet and other ontologies (Gonfa, 2018). In corpus-based approaches, a corpus that contains sense examples is prepared and the machine and deep learning algorithms identify the meaning of the ambiguous word in the context of text (Edi Faisal et al., 2018). In hybrid approaches, both knowledge-based and corpus-based approaches are combined to address their limitations (Andres Montoyo et al., 2005). As a researcher's best knowledge, there are no prior researches conducted regarding WSD using the approaches that were discussed above. As other languages, there are polysemy words in Wolaita language. This can be the cause of ambiguity problem for NLP applications that were developed by the prior researchers and also it can cause the problem of ambiguity for NLP applications will be developed.

As (Amlakie Aschale et al., 2021) claims, corpus-based approaches are recommended for languages that don't have WordNet and other knowledge-based dictionaries. In corpus-based approaches, three machine learning algorithms were applied for many languages. These are supervised learning algorithms, unsupervised learning algorithms, and semi-supervised learning algorithms. As we can see from the literature, supervised and semi-supervised approaches could identify only two senses for ambiguous words in sentences. In fact, one ambiguous word can have one up to sixteen meaning in the context of the text (Amlakie Aschale et al., 2021). In

addition to this, the unsupervised approaches could identify more than two senses for ambiguous words using a small amount of data. These can face the problem of the generalizability of the algorithm ( Teklay, 2018).

As stated by (Tesema, 2017), the performance of the corpus-based approaches increases using the optimal window sizes that refer to the presence of words from the left and the right of the ambiguous words. As many researchers claim, the optimal window sizes vary from language to language. But, the selection of optimal window sizes was based on two criteria as claimed by (Pinar Y, 2005). These are the accuracy achieved by the model and the computational time cost the model costs. As we can see from the previous work, the selection of optimal window sizes was only based on the average accuracy achieved by the model (Amlakie Aschale et al., 2021) (Getahun Wassie et al., 2014).

In this study, we aimed to conduct the WSD of Wolaita language using machine learning approach specially supervised learning approach by considering two main things discussed above. These are increasing the senses of ambiguous words in the context of sentences and selecting the optimal window sizes depending on the average accuracy achieved by the model and the computational time cost the model costs. As discussed above, due to the unavailability of WordNet, we manually prepared a corpus using language experts to conduct the corpus-based WSD of the Wolaita language.

WSD has many applications. In speech synthesis, WSD is important to determine the correct pronunciations of words to generate speech that sounds natural. This process is difficult for computers since there exist some words which are used in more than one context in the sentences. In MT, to be able to correctly translate a text, we need to know which sense is intended in the text. Therefore WSD required to identify the correct sense of ambiguous words in target language and source language. In IR, engines need WSD for filtering out documents with senses irrelevant to the query.

For the Wolaita language, no researches were conducted regarding WSD. But, the language has many polysemy words. These can be cause of challenges in NLP applications such as MT (Atnafu Lambebo et al., 2021), Text to Speech Synthesizer (Gebresilase, November 29, 2013), and Speech Recognition (Fanta, 2010) that were conducted for Wolaita language by the previous

researchers. Therefore, we argue that integrating WSD of Wolaita language with the above applications increases the performance of the NLP applications.

### 1.2 Motivation of the Study

The development of WSD for other languages increased the performance of NLP applications by identifying polysemy words in the language. There are a lot of polysemy words that are spoken by the Wolaita society. Therefore conducting WSD for Wolaita language can increase the performance of NLP application that have been developed for Wolaita language. In addition to this, the integrating WSD with NLP applications makes it easier for users to access the data by reducing ambiguity problems that occur in NLP. Therefore, these motivated us to conduct WSD for the Wolaita language.

### 1.3 Statement of the Problem

Nowadays, web technology and AI have been rapidly developing. These technologies highly use NLP in many aspects. For example, as stated by (Shubham Kumar et al., 2019), elderly people can control daily activities by feeding their sounds as input to the Smart Home system. And also one person can translate his/her language to his/her own target language by Google Translate. In addition to this, Chabot is used in the hospital to get information from patients (Siddique et al., 2021).

However, NLP is faced with an ambiguity problem, for computers can't understand the sense of the words from the texts that can have more than one meaning. Understanding the sense of the word in the text is easy for human beings but it is difficult for computers and it is an open problem in NLP (Mohammad Shibli Kaysar, 2019). Therefore, to solve these problems, many researchers conducted WSD for English, Chinese and other languages including Ethiopian languages such as Amharic, Afan Oromo, Tigrinya, and Geez (Bianca Scarlini et al., 2020) (Amlakie Aschale et al., 2021) (Reda, Meresa Mebrahtu, 2018) (Gonfa, 2018).

As the researcher's best knowledge, no prior work was done regarding WSD for Wolaita language. But, other NLP applications such as speech processing, MT, and Tex-to-Speech processing applications have been conducted for the language (Fanta, 2010) (Gebresilase, November 29, 2013). As stated by (Aung, 2011), integrating the WSD model with MT, speech

recognition increases the performance and the accuracy of the system. However, the absence of WSD faces the problem of ambiguity and performance for the applications that were mentioned above.

In linguistics, one word can have two up to sixteen meanings according to the context. As stated by (Amlakie Aschale et al., 2021), for each Geez ambiguous words, two senses were disambiguated and for total of six ambiguous words, 12 senses were disambiguated using ADTree and Bagging of semi-supervised learning algorithms. Similarly, ( Teklay, 2018) used ADTree and Bagging classifiers used disambiguate five ambiguous Tigrinya words, two senses were identified per sentences for three ambiguous words and three and four senses identified per sentences for the rest two ambiguous words and for total five ambiguous words, 13 senses were disambiguated. As stated by (Edi Faisal et al., 2018), two senses were identified for two ambiguous Indonesia words and total of four senses were identified per sentences. This indicates that highly polysemous words were ignored in previous studies and this can face still difficulties for NLP applications since more ambiguous words have more ambiguity problem for NLP. Some of the researchers tried to make the model to identify more than two senses per a sentences by clustering and a rule-based approach using a small amount of data. But this can reduce the generalizability of the model.

In addition to this, the effect of windowing was addressed only regarding the accuracy of the model but not the computational time of the model. And also, most researchers selected small window sizes. But, small window size can make the other ambiguity problem since it gives syntactic meaning but not contextual meaning. This indicates that selecting small window sizes and the large window size has negative effect on the performance of the model by (Daniel Jurafsky & James H. Martin, 2006).

### 1.4 The Research Questions

Based on the above points, our study answers the following questions.

**RQ1**: Which machine learning algorithms best perform for WSD of Wolaita language?

**RQ2**: How can the overall performance of WSD for Wolaita language be enhanced?

**RQ3:** To what extent does the model disambiguate ambiguous words from Wolaita texts?

## 1.5 Objective of the Study

### *1.5.1 General Objective*

The general objective of this research work is to design and develop WSD for Wolaita language using a machine learning approach.

### *1.5.2 Specific Objectives*

The specific objectives of this study work were listed as follows:

- To review the concepts and works related to WSD for other languages.
- To study the morphology and the formation of different kinds of ambiguities in Wolaita language
- To prepare a manually annotated corpus from different domains of data for selected ambiguous words of the language.
- To apply data preprocessing techniques for the Wolaita text.
- To design WSD model for Wolaita language.
- To develop a multi-class WSD model which best fits for Wolaita language.
- To evaluate the performance of the WSD model.
- To develop a prototype of WSD for Wolaita language

## 1.6 Scope and Limitations of the Study

### *1.6.1 Scope of the Study*

In this study, the WSD has been done for the sentence-level textual data that were collected from the four data sources. These are Holy Bible sentences, teaching materials from elementary schools and Wolaita language and literature Department, sentences gathered from FBC FM and Wogeta FM from Wolaita Sodo, and sentences collected from the Machine Translation Toolbox

that are freely available on the internet. We selected these sites based on their reliability and their usability by society. From these data sources, the sentences with ambiguous words were extracted and the five datasets were prepared based on context window size. Then the four supervised learning algorithms including SVC, RFC, Bagging Classifier, and AdaBoost Classifier with three feature extraction techniques including BOW, TF-IDF and Word2Vec were selected to disambiguate the ambiguous words from the sentences.

### *1.6.2The Limitations of the Study*

- Due to the unavailability of the WordNet and the time constraint, our study was limited to five ambiguous words but not all ambiguous words of Wolaita texts.

## 1.7 Significance of Research

WSD Model for the Wolaita language give significance for the society and the institutions by integrating with the following NLP applications.

***Machine Translation (MT)****:* WSD is required in MT for words that have different translations with different senses based on context (Mulugeta, 2020). For instance, in (Aung, 2011), WSD using Naïve Bayes classifier on Myanmar-English Parallel Corpus, it is indicated that the system improved the accuracy of Myanmar to English language translation system.

***Information Retrieval (IR)****:* WSD in IR is used as a prominent query formulation and expansion by eliminating documents containing the same word used with different meanings and also WSD is used to retrieve documents expressing the same meaning with different wordings. As stated by (Dian Paskalis, 2011) , the retrieval performance using query expansion increased by using WSD.

***Information Extraction (IE) and text mining***: WSD plays an important role for accurate analysis of text IE in different research works as Bioinformatics research (which is investigating biological issues using mathematics, informatics, statistics, and computer science) and Named Entity recognition system. Both of them include the task of analyzing meanings, this is how they are related to WSD.

***Speech Recognition*****:** In speech synthesis, the correct pronunciation of words used to generate speech that sounds natural is determined by the presence of WSD in Speech recognition.

***Institutions and Society*:** The development of the WSD Model for Wolaita language benefit Institutions like Wogeta FM, Wolaita TV, FBC Wolaita Soddo Branch, Wolaita Linguistics, Wolaita Language and literature department, and the Societies. These can be achieved when integrating it with the applications mentioned above.

***Computational Linguistics***: For the language owners, WSD helps as the computational linguists and it will help as contextual dictionary since in this time language linguists gets disappear (Girma, 2018).

***Semantically annotated Corpus:*** The researchers of WSD and NLP areas can access the datasets prepared in our study. Instead of preparing new datasets, they can increase the other ambiguous words and their corresponding sentences.

### 1.8 Organization of the Thesis

The remainder of the thesis is organized as follows: The second chapter presents the overview of WSD, the resources and the approaches used in WSD and general morphology of Wolaita language. In addition to these, it presents related works and the gaps identified. The chapter three presents the methodology used in our study. Chapter four illustrates the proposed solution of WSD model of Wolaita language. Chapter five describes the implementation of these proposed solution. Then Chapter six discusses the results and findings of the study. Finally, conclusion, recommendation and future work discussed.

# CHAPTER TWO

# 2. LITERATURE REVIEW

## 2.1 Overview of Word Sense Disambiguation and Natural Language Processing

In this chapter literature review in the field of WSD is reviewed and discussed in brief. The chapter covers an overview of NLP, WSD, resources used in WSD, and a discussion on major approaches that have been employed for WSD research with a special focus on a corpus-based approach that uses machine learning techniques. In addition to this, morphology of Wolaita language discussed. Finally, related works and their gaps are discussed.

***Natural Language Processing:*** It is sub field of AI, linguistics and computer science that is used to process and analyze large amount of natural language data. It also deals with the interaction of computers and human language.

There are three kinds of NLP[1] based on approaches used for processing the natural data. These are symbolic (rule-based) NLP (1950s-early 1990s), Statistical NLP (1990s-2010s), and Neural NLP (present). In symbolic NLP, the computers emulate NLU by applying rules to the data it is confronted from. In statistical NLP, machine learning algorithms are applied to address the hand-written rules used for NLP. It uses statistical inference to automatically learn from the corpora. In Neural NLP, representation learning and deep neural networks are used to process the large data and can achieve current-state-of-art result.

There are mainly seven common NLP tasks (Natural language processing, 2021). These are Text and speech processing, morphological analysis, syntactic analysis, lexical semantics, relational semantics, discourse semantics and high-level NLP applications. From those common NLP tasks, WSD is one of lexical semantics task that deals with semantics of a word in the context.

***Definition of WSD*:** In NLP, determining the sense of the word in the context is the open problem for NLP applications (Word Sense Disambiguation, 2021). But for human beings, this

[1] https://en.wikipedia.org/wiki/Natural_language_processing

is easy to understand the words used in text or spoken by a speaker. As defined by (Hundesa, Tesfa Kebede, 2013) words can have more than one meaning when they appeared in a particular text, and their meaning is identified by their context. As explained by Amlake (Amlakie Aschale et al., 2021), words may have two up to sixteen meanings according to their context. Therefore, we can say ambiguity in NLP is the ability to understand one word in more than one way. As defined by Scholarpedia (Philip & Eneko , 2008), WSD is the process of identifying the meaning of the ambiguous word in a particular context by activating the word. WSD is the NLP task and the automated process to recognize the sense of the word in a particular context.

Different types of ambiguities can occur in the text in terms of NLP. According to the explanation of (Anjali , M K1, 2014) (Wahab , Khan1, 2016), mostly occurring ambiguities are Lexical ambiguity, syntactic ambiguity, semantic ambiguity, anaphoric ambiguity, and pragmatic ambiguity.

***Lexical Ambiguity*:** Lexical ambiguity is one type of ambiguity that can be occurred in a single word. In this type of ambiguity sequence of tokens can have more than one meaning. For example, I saw a saw. The word 'saw' can be a verb or noun in this sentence. Therefore, 'saw' is the ambiguous word for the computer. This ambiguity can be addressed by WSD.

***Syntactic Ambiguity*:** This kind of ambiguity can give more than one meaning by the order of the word. For example, the sentence "We saw the boy with the camera". It is ambiguous whether we saw the boy carrying a camera or we saw him through the camera. This kind of ambiguity is addressed by POS Tagger. (NG'ANG'A, 2005)

***Semantic Ambiguity*:** Semantic ambiguity is one type of ambiguity that occurs when the words in the sentence can have misinterpreted meaning or the ambiguity occur when the sentence contains ambiguous words. For example, "The Tiger ate the sheep while it was running". In this sentence, there are two interpretations, "the first one is the tiger ate the sheep while the tiger itself running" and the second one is "the tiger ate the sheep while the sheep were running". This ambiguity is mainly addressed by WSD.

***Anaphoric Ambiguity*:** Anaphoric ambiguity can be occurred by anaphora entities or pronouns used in discourse. For example, "The plan was crashed by the hell. It was very steep. It soon burned. In these sentences, the anaphoric reference "It" occured ambiguity. That is, 'it' can

refer to both “plan” and “hill”. This is addressed by a machine-learning-based antecedent classifier (Yang, 2011).

***Pragmatic ambiguity*:** It refers to the situation where the context of a phrase gives it multiple interpretations. This is addressed by Domain Knowledge Graph (Alessio et al., 2014). The most pervasive ambiguity facing NLP today remains that of lexical ambiguity, where a word can have two or more associated meanings, depending on the context of use. WSD is the process by which contextual information is employed to resolve lexical ambiguity and determine the intended meaning (sense) of an ambiguous word (NG’ANG’A, 2005).

We can distinguish two variants of the generic WSD task (Hundesa, 2013):

***Lexical sample (or targeted WSD):*** In this task, WSD disambiguate restricted target words that occurred in the sentence. This kind of WSD uses supervised learning algorithms by training the machine by labeling the sense of ambiguous words by annotators.

***All-words WSD*:** In this task, WSD disambiguates all open classes in a particular text such as nouns, verbs, adjectives, and adverbs. In this way, the sense of the word is determined by the context in the document rather than one sentence. The sense of the word in the document is identified by word embedding (Rui Suzuki, 2018).

### 2.2 Brief history of Word Sense Disambiguation

The history of WSD research is old as MT. In the 1940s, Warren Weaver (Warren_Weaver, 2021), who was the first researcher found the problem of computational context which is faced in the machine during the early days of MT. According to early researchers, identifying the sense of the word could not be solved by the computer. Due to its difficulty in that period (Yehoshua Bar-Hillel, 2021).

In 1970, semantic interpretation systems that are sub part of WSD were developed with AI. The first semantics introduced was Wilks’ preference semantics. However, the problem in this period was the bottlenecks of the rule-base and hand-coded approach. Because this approaches consumes many times and resources and also they can be faced with knowledge acquisition bottleneck (Yehoshua Bar-Hillel, 2021).

In the 1980s, the knowledge-based approaches were appeared by the development of the Oxford Advanced Learner's Dictionary. Therefore, the problem of hand-coding was replaced by knowledge-based approaches. But, disambiguation was dictionary-based (Yehoshua Bar-Hillel, 2021).

In the 1990s, the evolution of the statistical machine learning approach and computational linguistics advanced WSD. For solving the problem of ambiguity, the three machine learning algorithms such as supervised learning algorithms, unsupervised learning algorithms, and semi-supervised learning algorithms used. But supervised learning algorithms reach a plateau and perform best than other algorithms (Word Sense Disambiguation, 2021) (NG'ANG'A, 2005).

In the year 2000s, other algorithms such as semi-supervised and unsupervised corpus-based systems combinations with different methods were developed and knowledge-based systems via graph-based methods got attention in the state of art. But still, supervised learning algorithms reach the plateau in accuracy (Word-sense disambiguation, 2021).

In Ethiopia, WSD research has been conducted for four languages such as Amharic, Afan Oromo, Tigrigna, and recently Geez language. The disambiguation of the ambiguous word is done by three approaches such as corpus-based approaches, knowledge-based approaches, and hybrid approaches that combine two knowledge and corpus-based approaches.

Among three approaches, the corpus-based approach was firstly attempted for the Amharic language by Teshome (Teshome, 2009). He demonstrated WSD by semantic vector analysis. Later, other researchers attempted many approaches to improve his work and advance the WSD of Amharic Language. Therefore, WSD of Amharic language was conducted on corpus-based approaches, recently knowledge-based approaches, and hybrid-based approaches. (Getahun Wassie and Solomon Tefera, 2014) (Yesuf, 2015) (Mulugeta, 2020).

Similarly, WSD has been conducted for Afan Oromo, Tigrinya, and Geez languages by using corpus-based approaches (Hundesa, 2013) (Gonfa, 2018). Later, development of knowledge sources such as WordNet, thesaurus, and other ontologies for Afan Oromo have been started (Yesuf, 2015).

## 2.3. Resources used in Word Sense Disambiguation

*A **machine-readable dictionary**:* It is an electronic dictionary that can be a single and multi-language explanatory dictionary and it is used when translation between two or more languages is done (Machine-readable_dictionary, 2021). It provides a ready-made source of information about word senses and knowledge about the world, which could be very useful for WSD and NLU. These approaches are used for exploiting the knowledge from the dictionaries, mainly used to avoid the need for large amounts of training data which is the bottleneck of the supervised learning approach.

***Thesauri:*** It provides information about relationships among words, especially synonymy. In the field of WSD, Roget's International Thesaurus is mostly and commonly used. Like MRDs, it containing information about relationships between words arranged in semantic categories representing different senses of words. However, thesauri provide a rich network of word associations and a set of semantic categories potentially valuable for large-scale language processing, it doesn't best fit for WSD because it has no rich information about word relations (Mulugeta, 2020).

***Ontologies**:* they are mostly used in computing and used for formal naming, the definition of categories, properties, and relationships between concepts and data entities that substantiate more than one domain of discourse. They are mainly used to represent the relationship between subjects by using their concepts and categories which represent the subject. In addition to this, they are arranged semantically rather than alphabetically (Mulugeta, 2020). This differs them from other knowledge sources that are described above. The most commonly used ontology is wordNet.

***WordNet*** is one type of ontology that is a large lexical database of English that was developed by Princeton University (Brown Keith et al., 2005) and it provides meaning for the word in the context by combining dictionary and structured semantic network with sets of cognitive synonyms that includes nouns, verbs, adjectives and adverbs that express a distinct concept.

In wordNet, words are organized into the conceptual structure by representing several semantic relationships such as hyponymy, hyponymy, meronymy, etc. among synsets (Yesuf, 2015).

Synsets are interlinked using conceptual-semantic and lexical relations. Words in the same synsets have similar meanings and can be used interchangeably without changing the meaning of a sentence. Each synset also contains glosses and examples for the concept.

***Unstructured Knowledge Sources*:** It is mainly corpus, whether it is labeled or unlabeled (raw). Among these sources, the brown corpus is a million-word balanced collection of texts published in the United States. In addition to this, BNC, WSJ corpus, American National corpus, and the Gigoword corpus are the widely known corpus. These unlabeled corpora are used for developing unsupervised WSD.

### 2.4. Feature-Based Word Sense Disambiguation and Context Window Size

The sense of meaning can be identified by the left words and right words from the target word. As stated in (Jurafsk, 2019), Feature-based approaches use context window size by which the left and the right words determine the sense of words. For example, if the window size is four, left and right words from the target word or ambiguous words are four.

### 2.5 Approaches used in Word Sense Disambiguation

There are three approaches mainly used in WSD in NLP tasks. These are knowledge-based, corpus-based, and hybrid approaches. The knowledge-based approach uses external knowledge resources like Thesaurus, WordNet, and soon. These approaches assign the correct sense for the word in the context by defining explicit sense distinctions. The corpus-based approach uses machine-learning techniques to disambiguate words according to their context. The hybrid WSD approach is a combination of both knowledge-based and corpus-based approaches. Due to the unavailability of WordNet and other knowledge-based resources, our study mainly focuses on corpus-based approaches that uses semantically annotated corpus to train machine learning algorithms.

#### *2.5.1 Machine Learning approaches used for Word Sense Disambiguation*

This methods use manually prepared semantic corpus to train machine learning algorithms. From these corpus, the senses of ambiguous words are induced. There are mainly three approaches that are used in machine learning approaches. These are supervised learning approaches, unsupervised learning approaches and semi-supervised learning approaches.

#### *2.5.1.1 Supervised Learning approaches used for Word Sense Disambiguation*

Supervised learning approaches use the labeled data to identify the sense of ambiguous words in the particular context. As discussed by (Mulugeta, 2020) (Amlakie Aschale et al., 2021), these methods achieve high results than others. Despites, machine learning algorithms have higher accuracy and performance, they require a large amount of human intervention to annotate the training data. This is the main limitation of this approach as compared to the knowledge-based approach. In addition to this, the problem of knowledge acquisition bottleneck has still faced difficulties for supervised learning algorithms (Andres Montoyo et al., 2005). It is estimated that to obtain the highest result at least 3700 words should be tagged with about 1000 occurrences each and the necessary effort to move thus is estimated to be 16 person-years (Gerard et al., 2000).

As solution to the knowledge acquisition bottleneck, four methods used in supervised learning algorithms to select training examples. The knowledge acquisition bottleneck of supervised learning algorithms has been pursued by four selection of training examples as stated by (Eneko Agirree and Philip Edmonds, 2007). These are the automatic acquisition of training examples, Active learning, Learning from labeled and unlabeled instances, and combing training examples from different words.

In the automatic acquisition of training examples, wordNet and other sense tagged corpus is used to acquire sense examples. In the second method which is Active Learning, there are two types of learning. These are Membership queries and selective sampling. In Membership queries, the learner constructs the examples and asks the teacher to label them. But this approach is difficult for WSD. In selective sampling, the learner selects informative examples from unlabeled data by measuring the amount of uncertainty in classification.

However, measuring the amount of uncertainty became a bottleneck for this method, and committee-based sampling which randomly derives several classification models from the training set, and the degree of disagreement between them is used to measure the informativeness of the examples.

The third method is combing training examples from different words by Open Mind Word Expert and sense examples are acquired from different data sources. The reliability between

annotators is measured by inter-raters agreement (IRR). For our research, we have used these method to acquire sense examples and their senses. We did it due to the unavailability of WordNet and other sense-tagged datasets.

The fourth one is learning from both labeled and unlabeled data by which few labeled sense examples are acquired and many unlabeled sense examples are acquired and a semi-supervised learning approach is applied.

Some of the machine learning (ML) techniques applied in supervised learning are discussed here.

***Naïve Bayes:*** Naïve Bayes classifier is a probabilistic method that assigns class for a sample based on Bayes theorem. To determine sense for a word w sense having maximum probability p (w=Si/f1,f2…..fn) is selected using each sense Si, and features that come with the context of w are chosen. There are mainly three types of naïve Bayes classifiers. These are Multinomial naïve Bayes, Bournoulli naïve bayes and Gaussian naïve Bayes. Multinomial naïve Bayes is used for text and document classification. On the other hand Gaussian naïve Bayes classifier is used for classifying continuous data and Bournoulli naïve Bayes are used for classifying Boolean data as explained by (Naive Bayes classifier, 2021). The formula of naïve Bayes is depicted in Equation (2).

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)} P(c|x) \;=\; P(x1|c)x \;\; P(x2|c)x \ldots P(xn|c)x \; P(c) \qquad \textbf{(2)}$$

***Decision tree:*** A decision tree is a prediction-based model which uses rules to partition the training dataset and to decide senses. As stated by (Jumi Sarmah et al., ,2016), the way the decision tree disambiguates the ambiguous words has its principles. Given a set of training samples with their sense category, we need to apply a mathematical function to get the best splitting attribute that determines the remaining training sample that is partitioned into several parts. A recursive procedure is followed in each partition to form a decision tree. If there are no remaining attributes on the tuples and all the tuples belong to the same class label (sense), the recursive partitioning stops. Then finally, majority voting is applied with the majority class in the sample. For example, for the noun sense of the word “Doona” in the sentence “Taani Wolayttatto doonaa haasayas ” that is “I speak Wolaita language” in English. The decision tree

is traversed to select the correct sense Doona/Wolayttatto in this context using the formula of Entropy in Equation (3).

$$Entropy = \sum_{i=1}^{c} -pi \cdot \log2(Pi) \quad \textbf{(3)}$$

***Exemplar-Based approach (K-NN):*** The exemplar-based approach works by selecting K nearest (most similar) neighbor example senses for an ambiguous word which is a measure of the smallest hamming distance. It is memory-based learning because training examples are kept in memory during the training phase. The value of K is known experimentally and if it is one of the test instances it will be assigned to its nearest neighbors. As it is indicated before, the similar vector means the nearest neighbor, so most of the time this distance is calculated as Euclidean distance between the instances to be classified to each of the examples in the training set (Mulugeta, 2020).

***Support Vector Machine (SVM):*** SVM is both a binary and multi-class classifier algorithm and which uses a hyper plane that is learned from the training examples. The hyper plane separates positive examples from negative examples by maximizing the distance between closest positive and closest negative examples which are known as support. So the main goal is to maximize this distance and minimize classification error. To make SVM fit the WSD problem, each sense of a word is considered as one class and the other remaining senses will be considered as members of the same class.

***Random Forest Classifier (RFC):*** It is one of the ensemble learning methods for classification and regression. RFC used multiple decision trees during training time. In the case of WSD, the output of RFC is the sense of meaning for Wolaita text selected by most trees in the algorithm. This classifier was used to solve the problem of overfitting that occurred in Decision. In RFC, the input vector which contains sense examples is fed to the algorithm, and each Decision tree votes for the class, and finally the highest number of votes especially the sense for ambiguous word is selected.

***Bagging*****:** It is one of ensemble method that works by combing the sum of each weak learner's weight to improve the stability and accuracy of the algorithms. The combination of weak learner mainly DT classifiers help to avoid overfitting and variance that can occur in the algorithm.

Although it is usually applied to decision tree methods, it can be used with any type of method (Bootstrap aggregating, 2021). As defined by (Getahun Wassie et al., 2014), it is a sample-based statistical method and it only needs a few samples instead of the large samples. It uniformly sampling instances with replacement.

***AdaBoost Classifier*:** It is also one of the ensemble methods that works by combining the weighted sum of weak learners to enhance its performance. As its name indicates, the AdaBoost classifier is adaptive by which weights of weak learners especially Decision Tree are re-assigned to each instance. For example, inaccurate weight in DT one sent to DT two, then DT two re-assign weights of the first DT. This process goes on until the model achieves higher performance.

#### *2.5.1.2 Semi-supervised Machine Learning Approaches*

These approaches are mainly appeared to solve the bottleneck of supervised learning approaches by reducing the large amount of tagged data required for supervised .when unlabeled data is used with a small quantity of labeled data increases the machine learning algorithm efficiency and gives improved performance with less effort. The common semi-supervised learning methods are discussed below.

***Bootstrapping algorithm*:** This algorithm was first proposed by Yarowsky and it was the first most successful algorithm which relies on a small number of labeled instances. In this algorithm, the initial classification model is learned from the instances using one of the supervised algorithms, especially decision lists. Then unlabeled instances are classified iteratively by the learned model and added to the labeled dataset. The training sets used in each iteration will be classified with confidence above a certain threshold will be used to further classify untagged sets for the future. The main advantage is the ability to increase training examples in every iteration from a small amount of initial training data. This iteration stops when no change is observed from the previous iteration (Mulugeta, 2020) (Amlakie Aschale et al., 2021).

***Bilingual bootstrapping methods:*** Bilingual bootstrapping methods works on words to be translated to other language using a small amount of labeled and large amount of unlabeled data for both target and source languages (Mulugeta, 2020). This is the only thing that makes it different from Yarowsky's bootstrapping. At every step, a classifier is constructed for both

languages and unclassified data will be added to the classified data for both languages. In constructing the classifier we can use one of the languages because words in one language have translations to the other. When evaluated on word translation disambiguation it outperforms monolingual bootstrapping.

***Label propagation algorithm:*** As mentioned by (Mulugeta, 2020), the Label propagation algorithm is the graph-based algorithm with labeled and unlabeled examples are vertices and edges as label information. If two vertices (examples) are closer they will have similar labels. But this is not determined only by labeled examples; closer labeled examples can also determine it. This algorithm works by global consistency assumption which makes it different from bootstrapping algorithms which are based on local consistency assumption. Local consistency assumption is based on the assumption that examples lose to labeled examples within the same class, whereas global assumption assumes that similar examples should have similar labels.

#### *2.5.1.3 Unsupervised Machine Learning Approaches*

Unsupervised machine learning algorithms used in WSD are called word sense induction by which sense for ambiguous words is induced automatically from the instance of words. According to the algorithm of Schutze (Jurafsk, 2019), the sense for the word is extracted by clustering algorithm as follows:

- Compute a context vector Vc for each token of Wt
- Use a clustering algorithm to cluster these word-token context vectors c into a predefined number of groups or clusters that each defines a sense of Wt.
- Compute the vector centroid of each cluster. Each vector centroid sj is a sense vector representing that sense of Wt

These approaches are mainly used to overcome the limitations of supervised learning algorithms. That is, it is a difficult and time-consuming method to get the required resources. Although unsupervised learning algorithms overcome the problem of supervised approaches, it has lower performance than other approaches. In addition number of clusters may differ from the actual number of senses, so this makes performance evaluation difficult. Consequently to check the quality of clusters humans must involve looking for the relationships between the members of clusters (Mulugeta, 2020)

There are two main approaches in the unsupervised approaches. These are distributional and translational equivalence approaches.

***Distributional approaches*:** They work on discrimination and are based on words that co-occur in similar contexts have similar meanings. In addition to this, large unlabeled corpus and graphs which describe how the senses are related are used (Johansson et al., 2015) (Eneko Agirre and Philip Edmonds, 2007). On the other hand,

***Translational equivalent approaches**:* They use parallel bilingual corpora of two languages which can be used for the automatic construction of sense inventory.

There are also two methods in the above approaches to induce sense for the words. These are token-based and type-based methods. In the distributional approach, Token based method works by clustering contexts in which the target word occurs with the same sense to one group. On the other hand type-based method in the distributional approach identify and cluster words that are to be co-occurring together with similar contexts. Whereas in translational equivalent, the type-based method produces a set of related words in the source language (bilingual dictionary) and the token-based method produces sense tagged text by giving an appropriate translation for a sense of target word for each of its occurrences (Mulugeta, 2020) (Pedersen, 2007). The common algorithms used for unsupervised WSD are discussed as follows.

***Context clustering:*** In this method, target words are represented by a context vector which is constructed for each occurrence of the context vector, and the vectors are finally classified. A vector includes all senses of a word and it is represented by average agglomerative clustering (Mulugeta, 2020). This set of vectors for a word is used to construct a co-occurrence matrix which is going to be used for similarity calculation between words. In constructing the matrix if we are dealing with a large number of dimensions Latent Semantic Analysis (LSA) is used to reduce it, then the similarity between words is calculated using cosine similarity. Finally, clustering is done based on the context similarity.

***Word clustering*:** In this method, clusters of words are formed rather than contexts. First words which are similar to the target word are identified. The similarity is based on Information Content based on a single feature, where having the highest IC is more similar and less similar if the IC is low. The listed words represent different senses of the word so clustering of these

words is performed at the last to classify them into senses. To do the clustering there is Latent Semantic Analysis, Hyperspace Analogue to Language and clustering By Committee algorithm (Mulugeta, 2020).

***Translational equivalence*:** It is the multilingual approach of WSD that is used in translation. It contains words aligned in parallel corpora of two languages. The disambiguation process was done firstly generating translation equivalent candidates and then it extracts the most similar translation equivalence pairs that consist of features used for translation. In this way, the appropriate translation of the target word is achieved (Mulugeta, 2020).

### *2.5.2 Knowledge-based Approach*

A knowledge-based approach is an approach that is used to make extensive use of knowledge sources that are described above to decide upon the senses of words in a particular context. As explained by (Word Sense Disambiguation, 2021), the senses of words are determined by a commonly used algorithm called the Lesk algorithm. When compared to corpus-based approaches, it doesn't require a large amount of sense annotated data as supervised and unsupervised approaches. But it has some limitations. As stated by (Mulugeta, 2020), it has poor performance and lower precision because it highly depends only on dictionary-based senses.

In Knowledge-based approaches, disambiguation was done by finding pair of dictionary senses with the highest word overlap in their definition in the dictionary. There are two methods used to find definitions. These are considering general word-sense relatedness and calculating the semantic similarity of each pair of word sense in WordNet. Nowadays BabelNet is also used as the knowledge-based resource in which a multi-lingual lexicalized semantic network that it provides the senses of content words, the semantic relationship between the senses, and the set of synonyms of the sense (Dongsuk O, 2018). The BabelNet uses the graph-based method and similarity-based method. In the similarity-based method, each sense of ambiguous words is compared with the surrounding context. The word with the highest similarity gains correct sense where as in the graph-based method, the node of the graph represents word senses and the edge represents meaningful relations.

### *2.5.3. Hybrid Approach*

A hybrid approach is a combination of corpus-based and knowledge-based approaches. The main aim of this approach is to get the advantage of having more knowledge sources and the strength of different approaches. Knowledge-based approaches can help corpus-based approaches when there is a lack of training data and corpus-based methods help in achieving high-performance results (Bekele, 2016) (Mulugeta, 2020) (Amlakie Aschale et al., 2021).

## 2.6 Sensival Evaluation

The Sensival evaluation is one of the evaluation methods of semantic analysis and WSD. The evaluation is done by the SemEval[2] workshop. The major goals of this evaluation are first, to identify whether the intended meaning of the word in the semantic analysis is at word level or not. In the WSD, the intended meaning of words is contextual. Second, how semantic analysis fits together with NLP applications such as IR, MT, speech recognition, and soon. Since semantic analysis plays a significant role in the performance of these applications. Based on Sensival evaluation, WSD has three phases:

- ✓ In the training phase, the training dataset should induce the sense inventories for the polysemy words. No resources allowed other than POS Tagging, morphological analyzer and syntactic parsers.
- ✓ In the test phase, the test data should be taken from the training set
- ✓ In the evaluation phase, the testing phase is evaluated by supervised and unsupervised learning.

## 2.7 Morphology of Wolaita Language

### *2.7.1 Overview of the Language*

Wolaita is one of the North Omotic language spoken groups in the Wolaita Zone and some other parts of the SNNP Region of Ethiopia (WAKASA, 2014). Wolaita has existed in written form since the 1940s when the Sudan Interior Mission first devised a system for writing it by Latin script. Now, the language has been advanced and many books, dictionaries, and also applications like Wolaita Holy Bible app and Wolaita English Dictionary app developed and in

[2] https://en.wikipedia.org/wiki/SemEval

addition, the language is being taught by Arbaminch Teaching College in Diploma Level and Wolaita Soddo University in BA Degree level.

### *2.7.2 Alphabet and Writing system of Wolaita Language*

The writing system of the Wolaita language is straightforward which is designed based on the Latin script. And the text is written from left to right and spaces between words use as demarcation.

### *2.7.3 The phonology of Wolaita Language*

Phonology is a branch of linguistics that studies the physical sounds of human speech and is concerned with the physical properties of phonemes, and the processes of their physiological Production (Fanta, 2010). The Wolaita language contains 34 phonemes (Morphology of Wolaita, 2021). These phonemes are categorized into vowels, consonants, and paired phonemes. In addition to this, there are eight paired phonemes such as[sh,ph,zh,ny,nh,ch,ts,dh] . As stated by Wakasa (WAKASA, 2014), the Wolaita language has 29 phonemes. These are mentioned below.

*Table 2. 1 Phonemes of Wolaita Language (Fanta, 2010)*

| ***Vowels(voiced)*** | ***A*** | ***e*** | ***i*** | ***o*** | ***u*** |
|---|---|---|---|---|---|
| Consonants(voiceless) | Voiceless stops | p | t | k | |
| | Voiceless fricatives | s | sh | h | nh |
| | Voiceless affricate | ch | | | |
| Consonants(voiced) | Voiced stop | b | d | g | |
| | Voiced fricatives | z | zh | | |
| | Voiced affricate | j | t | | |
| | Voiced nasals | m | n | | |
| | Voiced liquids | r[r] | l | | |
| | Voiced approximates | w | y[j] | | |
| | Voiceless glotalized consonants | *P (ph)* | *T (x),* | *K (Q)* | *C (c)* |
| | Voiced glottalized consonants | *D(dh)* | l | m | N |

### *2.7.4 Gemination of vowels and consonants, Pitch and Diphthongs*

In the Wolaita language gemination of consonants and vowels changes the meaning of the word. The gemination of vowels makes long and short word sound. For example, two words are different in their meaning.

a) **Mata** means 'nearby'. In this word, the vowel 'a' is not geminated.
b) **Maataa** means 'grass' and in this word vowel 'a' geminated like 'aa' before and after't'.

The second gemination of phonemes is the gemination of consonants which makes the geminated phoneme be sounded whether tightly or not. For example: The word '**Maataa**' and '**Maattaa**' are different words and in the first word 'maataa' the phonem 't' not geminated and the 't' is not sounded tightly. And the second word 'maattaa' means 'milk' in English. In this word, 'sounded tight.

***Pitch in Wolaita Phonemes*:** Pitch refers to the height and depth of the sound of phonemes. As stated by (WAKASA, 2014), the pitch of the phonemes may change the meaning of the words. For example, the words 'maataa' and 'dooriis' have a different meanings. The meaning of the words according to their pitch is as follows:

- ✓ **Máátaa** translated 'authority' in English
- ✓ **Maatáá** translated 'grass in English
- ✓ **Dooríís** translated 'he picked up in English
- ✓ **Dóóriis** translated 'he chose' in English

***Diphthongs*:** The combination of two vowels in a single syllable forms sounds of diphthongs. Wolaita has four diphthongs: /ai/, /au/, /oi/, and /ui/ as stated by (Fanta, 2010).These is mostly occurred in Wolaita Holy Bible. The following are examples of words that contain these diphthongs with their phoneme equivalent.

***ai***ll-íya ' ai ll i y a'
up***ái***tt-iis ' u p ai tt ii s'
***au***-g-áá ' au g aa'
dad-***áu*** 'd a d au

In other academic books like primary school, high school, and university, diphthongs are replaced as follows.

a) “Aifiya” replaced by ‘ayfiya’. In this word ‘i’ replaced by ‘y’
b) “Augaa” was replaced by ‘awugaa’. In this word ‘u’ is replaced by ‘wu’

### *2.7.5 Morphology of Wolaita Language*

Most words in Wolaytta consist of a lexical stem and a grammatical ending. As stated by Habtamu,Wakassa and Tedros (Fanta, 2010) (Gebreselassie, 2018) (WAKASA, 2014), the lexical stem has the only suffix but not prefix. For example:

*Table 2. 2 Suffix formation of Wolaita Language*

| ***Word*** | ***Stem*** | ***suffix*** |
|---|---|---|
| Keexiis | Keexa | -iis |
| Keexissiis | Keexa | -issiis |
| Keexidoogaapekka | Keexa | -idoogaapekka |
| Keexisisidaakaappekka | Keexa | -isisidaakaappekka |

***Syntax in Wolaita language:*** Wolaita language used SOV agreement for sentence formation. Modifiers precede their modified heads, but the use of appositive construction may obscure that. In the following, the secondword *lo"-uwa* is appositive to the first one *boyn-aa.*

*A Boyn-aa lo"-uwa shamm-aasu means that ‘she bought good taros’*
In the above sentence, She is subject, Boy-naa is subject, lo’’uwaa is modifier and shammaasu is verb.

### *2.7.6 Punctuation Marks in Wolaita Language*

Words in Wolaita texts are separated by white spaces the same way as it is used in English. Different punctuation marks follow the same punctuation pattern used in English and other languages that follow the Latin writing system. For example, comma (,) is used to separate listing of ideas, concepts, names, items, etc. and the full stop (.) in the statement, the question mark (?) in interrogative and the exclamation mark (!) in command and exclamatory sentences mark the end of a sentence.

### *2.7.7 Ambiguities in Wolaita Language*

Wolaita language has five types of ambiguities. These are Phonological Ambiguity, Lexical Ambiguity, Structural Ambiguity, Referential Ambiguity, Semantic Ambiguity, and Orthographic ambiguity. These are discussed below one by one with examples:

***Phonological Ambiguity*:** It occurs when words have the same sound but have different meanings and it is caused due to placement of pauses on structures. The difference in placement of pauses and their absence causes categorical and meaning differences. For example,

a) [**mata + asa** ]
b) [**matasa** ]

Sentence (a) is interpreted as "the closest person" when there is a pause (+) in [mata + asa]. But the absence of pause in [matasa] as in sentence (b) the sentence is interpreted as "Nearby place".

***Lexical Semantic Ambiguity*:** Lexical Ambiguity is a sub-type of semantic ambiguity that occurs at the word level and the word or lexical items may have more than one meaning according to the context. This ambiguity can be occurred by two things discussed below.

***Categorical ambiguity*:** This is caused by lexical elements having the same phonological form but different word classes or categories. For example, 'Wolaitan **masqqalay** keehippe bonchchettes' is interpreted as:

a) **Maskal** holy day is celebrated very much in wolaita.
b) In wolaita a person who **celebrate** maskal day is honored.

The source of ambiguity is / **masqqalay** / which have a noun meaning 'kind of holy day' in the first sentence and a nominal or verbal meaning 'celebrating Maskal' in the second sentence.

***Homonymy:*** This type of ambiguity is caused due to lexical items having the same phonological form but different meanings. For example,

a) '**shempuwaa xayaas**' interpreted as "I can't breath"
b) '**shempuwaa xayaas'** interpreted as "I am so busy."

The source of ambiguity is / shempuwa / which have two meaning discussed above.

***Homophonous affixes*:** Homophonous affixes are one type of ambiguity that is caused when the same stem word is added with affixes having the same phonological form but that have a different meaning. For example, "Baassi ta oosuwan **yootettiis**., can be interpreted as:

a) Basa is rumored to have done what I did.
b) Basa was angry for what I had done.

The source of the ambiguity is the suffix /**-ettiis**/ which may mean "rumored" or "angry" when added to the word /**yoot**/.

***Structural Ambiguity*:** Structural ambiguity is the most common type of ambiguity in the language which is caused because of sentences having more than one possible arrangement or syntax. For example, 'Ta keettaa **goday** anjjiis' and 'Ta **goday** keettaa anjjis.'can be interpreted as:

a) 'The owner of my house blessed me.
b) 'The Lord blessed my house'

This kind of ambiguities are addressed by POS taggers (NG'ANG'A, 2005).

***Referential Ambiguity*:** It is also a kind of ambiguity occurred when pronouns consist of a particular antecedent with a different meaning. For example, 'Na'ay ba aawaa bonchchiis. I keehippe ufayttiis' can be interpreted as:

a) The boy honored his father. Therefore, boy was pleased.
b) The boy honored his father. Therefore, the father was pleased.

As we can see from the interpretations '**I'** can refer to the **boy** and the **father** in English.

***Semantic Ambiguity*:** Semantic ambiguity is a type of ambiguity caused by words having differently related and unrelated meanings. These ambiguities are caused by polysemic, idiomatic, and metaphorical constitutes.

Polysemic constitutes: a word is polysemy if it has different senses but related in meaning. For example, '**Ufayssi xayiis**' can be interpreted as:

a) There is no happy.
b) Ufayssi (a person) disappeared.

The ambiguity is because of / Ufayssi / which may refer to 'happy' or a male person with the name 'Ufayssi'.

***Idiomatic constitutes***: It is done when there are words that can be interpreted in different way from the literal meaning. For example, '**Keettay maanaanes**'can be interpreted as:

a) The house is scary.
b) The house may eat us.

The ambiguity is because of / Manaanes/ which may refer to 'scary' or non-material behavior which is the house may eat.

To conclude that Wolaita language morphologically rich language and words are formed by adding suffixes to the end of the stem word. The suffix elongation and gemination of vowels and consonants make the language morphologically rich. In the above sections, we saw that how ambiguity occurs in the wolaita language. In this study, we focused on **lexical semantic ambiguity** that is mostly addressed by WSD (Anjali & Babu , 2014) and also other related works ( BEKELE, 2016) (Edi Faisal et al., 2018) focused on it. As discussed in section 2.1, the ambiguities discussed above addressed by their own solutions in NLP.

## 2.8 Related works

Many kinds of research were conducted in English, European, and other languages for WSD. Most European languages have enough resources for NLP tasks. Therefore, many types of research have been conducted to solve many NLP problems using knowledge-based approaches with the integration of graph-based models and deep-learning algorithms (Udaya Raj Dhungana et al., 2015) (Bianca Scarlini et al., 2020) (Lu, 2019).

In Ethiopia, WSD research has been conducted for four languages such as Amharic, Afan Oromo, Tigrigna, and recently Geez language. As the Ethiopian languages are under-resourced, most of the researches were conducted using corpus-based approaches. Recently knowledge-based approaches have been used for Afan Oromo and Amharic languages because of the construction of wordNet (Getaneh, 2020) (Gonfa, 2018) (Mulugeta, 2020). For the Wolaita language, we have used corpus-based approaches due to the unavailability of WordNet and other lexical dictionaries. Manually constructing WordNet requires language experts, NLP experts, much data, much time, and money. Therefore the related works discussed below were conducted by using a corpus-based approach.

Edil Faisal (Edi Faisal et al., 2018) conducted WSD for Bahasa Indonesia using supervised learning algorithms. In this work, they collected data from a Wikipedia article to find the sense of two ambiguous words such as ‘motor’ and ‘bulan’. They used TF-IDF and SVD algorithm for feature selection and classification algorithms such as SVM, multivariate Bournoli naïve Bayes and Multinomial naïve bayes. Then finally TF-IDF with SVM achieved the highest accuracy. But in this work, only two ambiguous words and their corresponding two senses were identified.

Getahun Wasie (Getahun Wassie et al., 2014) conducted a WSD model for Amharic words by using semi-supervised learning algorithms. In their work, only five ambiguous words are selected and 1031 sense examples were used to determine the sense of words. He conducted his research to address the knowledge acquisition bottleneck of supervised WSD of Amharic which was conducted by Solomon Mekonin (Mekonnen, 2010). As its name indicates supervised learning requires high human intervention to the label training dataset. To solve this problem, three Semi-supervised machine learning algorithms such as Adaboost, Bagging, and ADTree are used to classify the sense of ambiguous words and identify the sense words, they used window size 3-3 and finally achieved accuracy 84.9%,81.25%,88.45% respectively by using Weka package. Among the three algorithms, ADTree achieved the highest score. But the disambiguation of words in their work is limited to only two senses and five ambiguous words and the optimal window size selected was based on only average accuracy but not on the computational time and the nature of meaning(syntactic or semantic)

Meresa Mebrahtu (Reda, Meresa Mebrahtu, 2018) conducted Tigrigna WSD using an unsupervised machine learning approach. In his work, he used four ambiguous words and corresponding sense examples to identify the sense of the word. The window size used to determine the sense of the word is window 10 and two up to four senses of words are used to cluster sense examples of ambiguous words. To cluster the sense of the ambiguous word, K-means, Hierarchical agglomerative, and Expectation-Maximization (EM) implemented in the Weka package. Among five clustering algorithms, EM achieved accuracy 83%. He conducted this research to overcome the bottleneck of supervised learning. But he used only four ambiguous words as the previous researchers. The optimal window size selected was based on

only average accuracy but not on the computational time and the nature of meaning (syntactic or semantic).

Workineh Tesema (Tesema, 2017) conducted WSD and Semantics for Afan Oromo words using the Vector Space Model. He conducted this research to find the sense of ambiguous word based on surrounding context using context window size ±2. The unique sense from each cluster for ambiguous words is identified using Vector Space Model and cosine similarity and 85% accuracy was achieved. But he used only one ambiguous word 'Bahe' and prepared the corresponding sense examples for one ambiguous word. And also window size two can make the sense syntactic rather than semantic.

Teklay M ( Teklay, 2018) conducted WSD for Tigrinya language using semi-supervised approach. He conducted WSD of Tigrinya language using ADTree and Bagging classifiers. He selected window one (1-1) for 1250 sentences and five ambiguous words to identify two up to four senses of the polysemy words. In his work, he only focused on the average accuracy achieved. In addition to this, smaller window sizes can also make the sense to be syntactic rather than semantic meaning.

Yehuwalashet B. ( BEKELE, 2016) conducted the hybrid approach for WSD of Afan Oromo by combining the unsupervised and rule-based approach to find the meaning of words in the surrounding context. The meaning of the ambiguous word is captured by window sizes ±1 and ±2. He found one up to five senses for twenty ambiguous words from a total of 150 sense examples. In his work, he used two clusters as partitional clustering (EM, K-means) and hierarchical clustering. Among the clusters, partitional clustering achieved the highest accuracy. The system yield 76.05% for the unsupervised machine learning approach and 89.47% for the rule-based approach. however, the hybrid approach yield higher accuracy than the unsupervised machine learning approach, it works for the scarcity of data, it consumes time for making handcrafted rules, and it is also used with small-size data. And also window size two can make the sense syntactic rather than semantic.

Recently, Amlake A. (Amlakie Aschale et al., 2021) conducted corpus-based WSD for Geez language. In his work, he used three corpus-based approaches such supervised, unsupervised and semi-supervised approaches, and compared their average accuracy. For the supervised

learning approach, he used SMO, Naïve Bayes, and Bagging. For the unsupervised learning approach, he used EM, Simple K-means, Farthest First, and Hierarchical clustering. And for semi-supervised learning, he used AdaBoost and ADTree algorithms. Among the three approaches, semi-supervised learning algorithms especially ADTree achieved 91.1% average accuracy. For training the machine learning algorithms, he used 2119 sense examples for six ambiguous words. To find the correct sense of the ambiguous word from the surrounding context, window size ±4 was used. As previous researches of corpus-based approach, the knowledge bottleneck is the limitation of this research and also he used only two senses for six ambiguous words for one word may have more than two senses (Amlakie Aschale et al., 2021). And also in data preprocessing, he developed a stemming algorithm for only six ambiguous words but not for all word classes.

*Table 2. 3 Summary of related works*

| ***Authors and year*** | ***Objective*** | ***Methodology/ies used*** | ***Limitations*** |
|---|---|---|---|
| Getahun Wassie(2014) | To find the sense of Amharic ambiguous words using semi-supervised learning. | ADTree on window size(3-3)<br><br>(88.45% accuracy ) | ✓ only two senses for five ambiguous words are disambiguated |
| Yehuwalashet Tessema(2016) | To find the sense of Afan Oromo ambiguous word in the context using clustering and rule-based approach. | EM combined with rule-based approach on window size(1-1)<br><br>(90.3% accuracy) | ✓ Window size (1-1) gives syntactic meaning rather than semantic meaning.<br>✓ He used a small amount of data and these my reduce generalizability of model. |
| Workineh Tessema(2017) | To find the meaning of Afan Oromo ambiguous word in the context using clustering similar contexts | Vector Space model on Window size(2-2)<br><br>(85% accuracy) | ✓ The model predicts more than two senses. But he used only one ambiguous word. |

| | | | |
|---|---|---|---|
| Meheressa Mebratu(2018) | To find the sense of Tigrinya ambiguous word in a context using clustering similar contexts | EM model on window size(2-2) achieved better accuracy<br>(83% accuracy) | ✓ The model predicts more than two senses with a small amount of data.<br>✓ This reduces generalizability of model |
| Teklay Muruts(2018) | To find the sense of Tigrinya five ambiguous word in a context using semi-supervised learning | ADTree and Bagging<br>Used on window size 1(1-1) | ✓ The window size 1(1-1) can make the sense syntactic. |
| Edi Faisal,et al(2018) | To find the sense of the Indonesia word in the context by classifying sentences with the ambiguous word | Used SVM<br>No window size discussed<br>TF-IDF used<br>(87.7 % accuracy) | ✓ the effect of window size not addressed<br>✓ the model predicts only two senses for two ambiguous words that words |
| Amlake Aschale(2021) | To find the sense for six ambiguous words in Geez language | ADTree and Bagging Classifiers with TF-IDF achieved better results on window size 4 (92.1 Accuracy) | ✓ The model only predicts two senses for ambiguous words. |

## 2.9 Summary of Related Works

In the above Table, the corpus-based WSD model was conducted using supervised algorithms, unsupervised algorithms, semi-supervised algorithms, and hybrid approaches(clustering and rule-based). Due to the unavailability of WordNet, researchers prepared datasets manually with context window size. For Wolaita Language, there are no prior researches done regarding WSD. The Wolaita language consists of a lot of ambiguous words. For example, ‘Naaga’ has the following English meanings. These are ‘protect’, ‘preserve’, ‘wait’, ‘keep’, ‘conserve’, and soon. Therefore, integrating the WSD model with NLP applications such as MT, Speech Recognition, and other NLP tasks have been developed for the Wolaita language (Fanta, 2010) (Gebresilase, November 29, 2013) (Atnafu Lambebo et al., 2021) which increases the performance of the model.

From the articles reviewed, we got gaps that were identified in the literature. In many languages, one word can have two up to sixteen meanings in the context (Amlakie Aschale et al., 2021). But in the above literature, only one up to two senses are mostly disambiguated. Some of the researchers tried to find more than two meanings for selected ambiguous words using clustering algorithms and hybrid approaches (clustering and rule-based). But the model was trained on small data. These affect the generalizability of the model. In addition to this, the effect of windowing was addressed only regarding the performance of the model but not the computational time the mode costs. Lastly, most of researchers used small window sizes to train the model. As stated by (Daniel Jurafsky & James H. Martin, 2006), shorter context windows tend to lead to representations that are a bit more syntactic rather than contextual. In addition to this, the larger window size yields more computation for the algorithm (Jurafsk, 2019).

# CHAPTER THREE

# 3. RESEARCH METHODOLOGY

## 3.1 Overview of Research Methodology

In this chapter, the methodology of the research, data collection and preparation, data preprocessing, feature extractions techniques, the algorithms and model evaluation techniques used to conduct WSD are discussed. In addition to this, tools used to conduct these research and the prototype of our model are discussed.

## 3.2 Research Design

To achieve the general and specific objective of our study, we used quantitative experimental research design to select the best and outperforming models and feature extraction techniques. In this research methodology, we collected data from four sources including Holy Bible available online, data repository of MT platform available online, data from media agencies specially FBC Wolaita Sodo branch and Wogeta FM. In addition to this, we collected teaching materials from Wolaita Language and Literature Department and elementary schools. We selected these sites to train the model with different domains of data and to increase the generalizability of the model. Then after data collection, datasets were prepared based on the context window sizes. Then after dataset preparation, data preprocessing techniques were applied to train the model with quality data. Since statistical machine learning algorithms require data in vector form, we transformed data using feature extraction techniques. Finally the machine learning algorithms including SVC, Bagging Classifier, Random Forest Classifier and AdaBoost Classifiers trained based on context window sizes. Then the models were evaluated by CV and performance evaluation metrics.

## 3.3 Data collection and Data Preparation

There are no WSD datasets prepared by prior researchers. Therefore, in this study data collection becomes the first issues rather than corpus preparation. In data collection and preparation, we set a criteria to attain the objective of our study in terms of accuracy. In machine learning

approach, there are significant issues to be considered to increase the performance of the algorithms. Because, pure data are soul of machine learning algorithms. Therefore, to attain these objective, we set criteria to collect and analyze data.

- To collect the data from the data sources, Wolaita Language and Literature Department academic staff members helped us to select mostly spoken ambiguous words based on the number of senses they do have in the context of text.
- In addition to this, we selected the ambiguous words based on their frequency in the four data sources. The ambiguous words with lower frequency in the data sources were ignored. Because, in machine learning algorithms biases decreases if equal amount of data distribution is done.
- Due to the time constraint, we selected only five ambiguous words and collected the sentences that only contains five ambiguous words. For each ambiguous words, we prepared their corresponding dataset.
- These polysemy words are '**doona'**, '**ayfiya'**, '**naaga'**, '**aadhdha'**, and '**ogiya'**. From these ambiguous words, the '**naaga'** has four senses and the rest have three senses in the context of text.
- Then we collected data from four sources. These are Holy Bible, News agencies, Academic data and freely available data on the internet that stored by the researchers. We selected these, to train the model with different domains of data. These increases the generalizability of the model ( Teklay, 2018).
- We collected only text data from the four sources such as Holy Bible from internet, News data (Sport, Business, Health, and national and international news) from Wogeta FM, and FBC FM Institutions, Academic documents from Wolaita language and Literature Departments and elementary schools, and data from free and open source MT platforms brought by researchers[3]. This indicates that the data gathered from reliable places or the internet.
- Then after collecting the data, we equally distributed the data points. In our case, data points are sentences with ambiguous words. These help the model to reduce the overfitting problem that can be caused by unequal distribution of data.

[3] https://sourceforge.net/p/apertium/svn/77956/tree/incubator/apertium-wal/texts/

- We selected four language experts to annotate the data collected from the data sources to ensure the reliability of data and training the model with quality data increases the performance of the model.
- We collected 4160 raw sentences from those data sources by manually searching the sentences with ambiguous words. Some of the data available on MT Toolbox Platform that are freely available online are similar with the sentences collected from teaching materials. Therefore, we removed duplicated data using MS Excel package. Some outliers mainly sentences that contain ambiguous words as suffix of other root words were also removed from our data. Because, only sentences with ambiguous words are selected based on the Sensival evaluation criteria of WSD.
- Finally, we gave 2797 cleaned sentences for the annotators

### *3.3.1 Sources of Data*

***Holy Bible***[4]**: We** collected data from Holy Bible written in Wolaita Language which is available online. The Bible Organization translated the Holy Bible written in English to the Wolaita Language. We gathered the bible data from the link mentioned in footnote below. We collected ambiguous words from the bible to include the meaning of the words in the bible perspective.

***Academic documents*****:** We collected academic data from elementary text books and Wolaita Language and Literature Department modules. These data sources contains not only academic contents but also poems, proverbs and other cultural issues of Wolaita Societies.

***Online Data Sources***[5]**.:** We also collected freely available data from the internet which was stored by the former researchers. These data were gathered from free and open source MT platforms brought by researchers.

***Data from Medias*****:** These data are national and international news, sports news, health and business news from Wogeta FM, and FBC FM news. Some of the news data collected from the Wolaita literature Department and other news are manually translated to the Wolaita language.

[4] https://ebible.org/study/?w1=bible&t1=local%3Awal&v1=JN1_1
[5] https://sourceforge.net/p/apertium/svn/77956/tree/incubator/apertium-wal/texts/

In addition to this, Wogeta FM and FBC FM radio stations use hand-written news in Wolaita Sodo. So we converted this data into an electronic document form by language experts.

We selected these four data sources based on the following sampling criteria.

- The Holy Bible and Academic data sources mostly used by the prior researchers (Fanta, 2010) (Gebresilase, November 29, 2013)
- These data sources are currently used by societies.
- These data sources are consistent and reliable. They can't be modified by the unauthorized personnel.
- These data sources are the main sources of textual data of Wolaita language. There are no Facebook and twitter data posted using Wolaita text.
- Information from the news agency is currently in use and is updated daily.

Based on these criteria, we selected these four data sources to enhance the generalizability of the model using different domains of data.

*Table 3. 1 Source of data and the amount of data collected*

| | ***Sources of Data and the number of data collected*** | | | | |
|---|---|---|---|---|---|
| ***Ambiguous words*** | ***Holy Bible versus*** | ***Academic data*** | ***Media*** | ***Other researchers*** | ***Total*** |
| **Doona** | 200 | 300 | 140 | 300 | 940 |
| **Ayfiya** | 250 | 200 | 150 | 120 | 720 |
| **Aadhdha** | 260 | 200 | 170 | 150 | 730 |
| **Naaga** | 300 | 250 | 100 | 120 | 770 |
| **Ogiya** | 250 | 400 | 200 | 150 | 1000 |
| **Total** | 1260 | 1350 | 760 | 840 | **4160** |

The collected data in the above tables are raw data. Some of the sentences in the collected data used the ambiguous words as suffix rather than words. In addition to this, some of the sentences with ambiguous words has few senses (meaning). Therefore we left these sentences and we selected the sentences with many senses (meaning) in the context of text.

### *3.3.2 Data Annotation*

Data annotation is the process of labeling or categorizing text data for AI applications. In our study, we selected annotators to keep the nature and behaviors of Wolaita language texts and to acquire quality and reliable data. The data annotation was done by four Wolaita Language and Literature Department experts based on their academic rank and skill to annotate data. We selected MSc holders and lecturers of Wolaita language. The annotation was done according to the annotation guideline that is stated in Appendix 1. And finally, three senses (meanings) labeled for '**doona'**, '**ayfiya'**, '**aadhdha'**, and, '**ogiya'** respectively and four senses labeled for the word '**naaga'** polysemy word. Totally 16 senses (meanings) of five ambiguous words were labeled for 2797 sentences. Table 3.3 shows, the data annotation result for each ambiguous word.

*Table 3. 2 The Data Annotation Result*

| ***Ambiguous words*** | ***No. of sense example*** | ***No. sense*** | ***No. of sense examples for each class*** | | | |
|---|---|---|---|---|---|---|
| | | | ***Sense 1*** | ***Sense 2*** | ***Sense 3*** | ***Sense 4*** |
| aadhdha | 500 | 3 | 174 | 167 | 159 | - |
| Doonaa | 500 | 3 | 172 | 164 | 164 | - |
| Ayfiya | 650 | 3 | 206 | 231 | 213 | - |
| Ogiya | 570 | 3 | 187 | 196 | 187 | - |
| Naaga | 577 | 4 | 126 | 144 | 150 | 157 |
| Total | **2797** | 16 | | | | |

### *3.3.3 Fleiss' Kappa and Inter-rater (Annotators) Reliability Measures*

The reliability of annotated data analysis was done by Fleiss' Kappa using SPSS Statistics software. We selected four raters (annotators) and we calculated the reliability measure to know whether they followed the annotation guideline or not. As stated by (Dehaibi et al., 2020), the Inter-Rater Reliability measure is used to enhance the quality of the model. In addition to this, the Wolaita Language and Literature Department confirmed the reliability of data annotated as mentioned in Appendix 2. The interpretation of kappa value k listed below in Table 3.2.

*Table 3. 3 Interpretation of Fleiss' kappa value*

| ***k*** | ***Interpretation*** |
|---|---|
| <0 | Poor Agreement |
| 0.01-0.20 | Slight agreement |
| 0.21-0.40 | Fair agreement |
| 0.41-0.60 | Moderate agreement |
| 0.61-0.80 | Substantial agreement |
| 0.81-1.0 | Almost perfect agreement |

To know the reliability of data and the overall agreement between the four annotators, we used Fleiss' Kappa and computed kappa (k) values and we got k=0.819 using SPSS Statistics software as mentioned in Appendix 3. This indicates that the overall agreement between the annotators is almost perfect. Therefore, we ensure that the data is reliable and can enhance the quality of our model.

### *3.3.4 Dataset Preparation and Unit of Analysis*

We manually prepared the dataset using the annotated sentence given by experts. This indicates that our model uses the sentences to identify the senses of ambiguous words. Despite the language experts annotated the data, they don't know about the preparation of datasets. Therefore, we prepared five datasets for the five ambiguous words based on the Sensival evaluation criteria. In addition to this, we prepared our data based on context window size. In our dataset preparation, the following things were performed.

- We used MS Excel to prepare, filter and clean the datasets
- Only the sentences with five ambiguous words were selected and others were omitted.
- Datasets prepared by equal amount of sense examples as discussed in section 6.2
- Ambiguous word (target) word put in the middle and others are put from the right and left of target word to train the model with the context window sizes. The main goal of window size is to capture the sense of ambiguous word from neighboring words by loading from datasets.
- The blank rows in the datasets be removed.

- Finally, we prepared five datasets as shown in Appendix 4.

### 3.4 Data Preprocessing

Data preprocessing is the techniques used to train the model with quality data. In this steps, we mainly used tokenization, stopwords removal, data cleaning, and normalization. In addition to this, we addressed missing values by replacing with zero.

***Handling Missing Values:*** In our study, we handled missing values by replacing with zero. For our datasets made based on window sizes, some words from the left and right of target words may be missed based on the morphology of the language. Therefore, these missed values replaced by zero using python.

***Data Cleaning*****:** The data collected from different sources usually contain special characters, punctuations, symbols, and Emojis. Therefore, data cleaning removes irrelevant things that can affect the performance of the model.

*Pseudo code 3. 1 Pseudo code for Data Cleaning (Nigatu, 2021)*

***Data Cleaning Algorithm***

**Input:** Input texts from datasets

**Output:** Clean Text

**INIT:**

**1. Read** the text in the dataset;

**2.While (**! end of the text in a dataset**):**

**If** url_regex = ('http[s]?://(?:|[0-9]|[$ _@.&+]|"[!*\(\),]|(?:%[0-9a-fA-F][0-9a-fA F]))+') **Then** replace by Space

**If** the text contains special char = "[@#$%^&=?×!,;:_.(){}`'/+*<>\"¤— „\ ®¯™¡¡\x10»€«·‘01§”¬¦…""f÷\~¨©±¥£¶–°•˜’“|]"] **Then** replace by Space

**If** a text contains emoji = [☺😑😊😷😵🤦👄] Then Replace by Space

**Return** clean_text;

**ENDIF**

**3.HALT**

***Tokenization*:** It breaks the raw text into words, sentences called tokens. It is used for interpreting the meaning of the text by analyzing the sequence of the words. Therefore, in this thesis, we have used tokenization to split sense examples into tokens. For example, "Ufayssi **ba aawaa ogiyan biis**." meaning 'Ufaisa followed in his father's footsteps' in English. This sentence is tokenized into'ufayssi','ba','aawaa','ogiyan','biis'.

***Stop Word Removal:*** The stop words are commonly used in texts in natural languages. But, in the NLP, they are not required for training the model. In data preprocessing, removing unnecessary words from the dataset increases the performance of the model. In the python NLTK package, there are sixteen stop words for sixteen languages. But for Wolaita, there is no stop word. Therefore, we used stop words for the Wolaita language that were collected by the previous researchers (Lessa, 2003) as shown in Appendix 5. The size of window size can be reduced if stopwords are removed. In our study, we prepared sentences based window sizes to load sentences from the datasets. The removal of stopwords can't change the position of the ambiguous words since our datasets are prepared by MS Excel.

*Pseudo code 3. 2 Pseudo code for Stopword removal (Nigatu, 2021)*

***Pseudo code for Stop Word Removal***

**Input:** Input texts from datasets

**Output:** removed stopwords

**INIT:**

**1. Read** the text in the dataset;

**2.For** (each word in a sentence):

If a word is in stop word list Then Remove a word

End if

**End for**

**3.Return** the remaining words

**4. HALT**

***Normalization*:** In NLP, normalization is transforming the texts into the standard form. As discussed in section 4.3, data was collected from different sources. Some words in these data are similar in meaning but different in spelling. For example, the word '**Dotoriya'**, '**Dr**.' and '**Dokttoriya'** are similar in meaning but they are differently spelled words. In addition to this,

in Holy Bible, some suffixes of the words are spelled differently when we compared them with academic text. For example, '**naaganaw', 'na7aa',** and **'aifiya'** are spelled differently in academic texts as '**naaganawu'**, '**na'aa**', and '**ayfiya'** respectively. Therefore we normalized these kinds of words by replacing '**w'** by '**wu'** and '**i'** by '**y'** and '**7**' by apostrophe " **'** " based on the current writing standard mainly used by Wolaita Language and Literature Department. The normalization of Wolaita words discussed in pseudo code 3.3 below.

*Pseudo code 3. 3 Pseudo code for WSD Normalization (Nigatu, 2021)*

***WSD Normalization Algorithm***

***Input:*** *WSD dataset*

***Output:*** *normalized WSD dataset*

***INIT:***

1. ***Read the WSD dataset***
2. ***WHILE (it is not the end of file):***

    ***IF*** *word ends with [ai], THEN* ***replace*** *with 'ay '*

    ***IF*** *word ends with [oi], THEN* ***replace*** *with 'oy '*

    ***IF*** *word ends with [aw], THEN* ***replace*** *with 'awu '*

    ***IF*** *the words ends with 'daana'* ***replace*** *by 'de'ana'*

    ***IF*** *the words ends with 'katamaa* ***replace*** *by 'allaana'*

    ***IF*** *data contains [Dr., Doc, Dr, Doktor] THEN* ***replace*** *with 'Dotore'*

    ***IF*** *data contains [Pr., Prof., Proffesor, Proff.] THEN* ***replace*** *with 'Prof.'*

    *Return normalized text*

    ***ENDIF***

3. ***HALT***

## 3.5 Feature Extraction Techniques

Corpus-based approaches are one of the statistical machine learning approaches. In this approach, statistical and probabilistic calculations are applied to train the algorithms. In our thesis, we have used text data. So these data should be converted to numbers especially vectors. We selected TF-IDF because, as claimed by (Robert Dzisevič,Dmitrij Šešok, 2019), (Fauzi, 2019), and (Faiza & Mohammed , 2021), the TF-IDF feature extraction approach outperforms other methods such as Word2vec and BOW by allowing the classifier to achieve the highest

accuracy when working with larger datasets and smaller datasets. In addition to this, it ignores the frequent and irrelevant words in the datasets and gives priority for important features. In addition to this, many WSD types of research of the other languages used TF-IDF (Edi Faisal et al., 2018) (Amlakie Aschale et al., 2021) (Gonfa, 2018). To ensure the above reasoning, we conducted an experiment to select best feature extraction for WSD of Wolaita Language in section 6.2.

### 3.6 Model Selection

In this work, we selected corpus-based WSD for the Wolaita language due to the unavailability of WordNet. As discussed in chapter two, corpus-based approaches include supervised learning, unsupervised learning, and semi-supervised learning algorithms. In our study, we selected supervised learning approaches based on their performance achieved in Sensival Evaluation of WSD and these approaches reached plateau in accuracy when we compared with other approaches (Word-sense disambiguation, 2021). Unlike semi-supervised learning, supervised learning algorithms work well in both smaller and larger dataset. In addition to this, they work best by classifying more polysemous words in WSD (Osman, 2016).

We also decided to select supervised learning based on the size of our datasets. We prepared 2797 sentences for five ambiguous words. Unlike Deep learning, the performance of supervised learning algorithms don't degrade with smaller datasets.

From supervised learning algorithms, we selected multi-class classification algorithms performed best for WSD model other languages (Edi Faisal et al., 2018) (Berke Özen et al., 2017) (Amlakie Aschale et al., 2021) (WELEMICHAEL, 2018). In addition to this, most of selected algorithms are ensemble methods that can handle overfitting and data imbalance problems in the datasets. Based on these, we selected multi-class classification algorithms including SVC, RFC, Bagging and AdaBoost Classifiers.

***Support Vector Machine (SVM)****:* It is one of the binary classifier algorithms that use a hyperplane to separates training examples. The hyperplane separates positive examples from negative examples by maximizing the distance between closest positive and closest negative examples which are known as support. So the main goal is to maximize this distance and minimize classification error. As explained by (Mulugeta, 2020) (Eneko Agirre and Philip

Edmonds, 2007), SVM is a binary classifier and it classifies the samples into two and more than two classes. In the WSD model of Wolaita, there are more than two senses for ambiguous words. Therefore, SVC classifies the sense of ambiguous words in the context into more than two classes using OVR and OVO parameters used to classify multiple senses in WSD of Wolaita language.

***Bagging*:** It is one of ensemble method that works by combing the sum of each weak learner's weight to improve the stability and accuracy of the algorithms. The combination of weak learner mainly DT classifiers help to avoid overfitting and variance that can occur in the algorithm. Although it is usually applied to decision tree methods, it can be used with any type of method (Bootstrap aggregating, 2021). As defined by (Getahun Wassie et al., 2014), it is a sample-based statistical method and it only needs a few samples instead of the large samples. It uniformly sampling instances with replacement.

*Pseudo code 4. 4 Pseudo code for Bagging Classifier algorithm (Getahun Wassie and Solomon Tefera, 2014)*

***Bagging classifier algorithm***

***Input***: training set D, Inducer I and the number of bootstrap samples T as input.

1. For *i* =1 to *T*{
2. *S*' = Bootstrap samples from *S*'
3. *C' = I(S')*
4. }
5. C* = arg max $= \sum_{ci=y}^{n} \mathbf{1}$

***Output*:** Generate a classifier C*

***Random Forest Classifier (RFC):*** It is one of the ensemble learning methods for classification and regression. RFC used multiple decision trees during training time. In the case of WSD, the output of RFC is the sense of meaning for Wolaita text selected by most trees in the algorithm. This classifier was used to solve the problem of overfitting that occurred in Decision. In RFC, the input vector which contains sense examples is fed to the algorithm, and each Decision tree votes for the class, and finally the highest number of votes especially the sense for ambiguous

word is selected. RFC is also used to handle data imbalance problems (Mrs. A. S. More and Dr. Dipti P. Rana, 2017)

***AdaBoost Classifier*:** It is also one of the ensemble methods that works by combining the weighted sum of weak learners to enhance its performance. As its name indicates, the AdaBoost classifier is adaptive by which weights of weak learners especially Decision Tree are re-assigned to each instance. For example, inaccurate weight in DT one sent to DT two, then DT two re-assign weights of the first DT. This process goes on until the model achieves higher performance. The process of model re-assignment in weal learners is depicted in figure 3.1 (Great Learning Team, 2020). Like other ensemble methods, the AdaBoost Classifier is also used to reduce the overfitting problem.

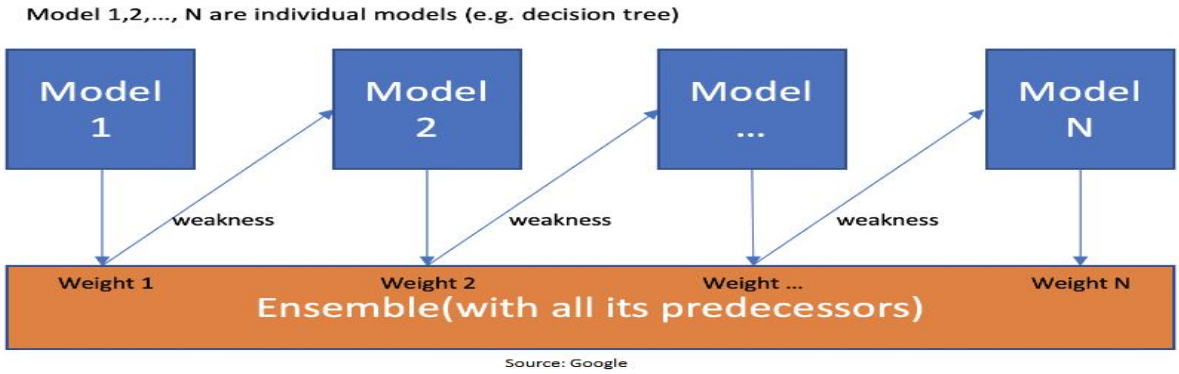



*Figure 3. 1 AdaBoost Classifier*

***Base Estimators (Weak Learners):*** As discussed in the above section, ensemble methods use weak learners and combine the sum of each weak learners' weight to enhance its performance. In the Bagging Classifier, by default DT is used as a weak learner. In this study, we selected additional three weak learners for Bagging Classifier to enhance the performance of the Bagging Classifier. The weak learners selected were Linear SVC Stochastic Gradient Descend, and Logistic Regression.

***Stochastic Gradient Descend (SGD) Learner***: It is one of the optimization algorithm used to minimize cost function by picking one data point from the whole dataset at each iteration. In another word, it updates the weight incrementally with the training sample. As a result, SGD updates gradient descend function until it reaches to lower gradient. This is used to reduce the cost of computation and also increase the performance of the model.

***Linear SVM Learner***: This classifier separates the data into classes using a hyperplane. Unlike the above Linear SVM, the sum of each weak Linear SVM classier combined to enhance the performance of the ensemble methods that are discussed above.

***Multinomial Logistic Regression Learner:*** This learner is a multi-class classification algorithm that generalizes logistic regression. It can predict more than two classes of data. In logistic regression, the sigmoid function is used as a cost function while in multinomial logistic regression softmax is used. In this study, it is used as a weak learner and the combination of it enhances the performance of ensemble methods.

***Decision Trees***: These are the default weak learner in the ensemble method. In DT, leaves represent class labels and branches represent the features. Information gain in DT is used to decide the features to be split. In our study, the combination of several weak DT learners combined to enhance the performance of ensemble methods.

***Regularization Techniques*:** We also equally distributed sentences in the section 3.3. In addition to this, we used the regularization technique to solve the overfitting problems that can occur in the model and also helps the model to predict unseen data.. The regularization technique used here is L2 regularization. We selected it based on the nature of our datasets. Words in our sentences are correlated and the combination of them gives the full meaning (semantics) of sentences. Therefore, setting these feature to be zero using L1 regularization not good for WSD. Instead of this, assigning small weights to the features by L2 regularization is important for correlated words (Ying, 2018). Based on these issues, we selected **L2 regularization.**

### 3.7 Model Performance Evaluation Metrics

The evaluation and testing of our models used two methods in this work. These are K-fold CV and performance evaluation metrics such as confusion matrix, accuracy, precision, recall, and F1-score. We selected K-fold CV for the following reasons. These are, the ability of the model to predict unseen data increases in K-fold CV, the problem of overfitting reduces by CV. And also, it is a popular method of evaluation. Therefore, we selected K-fold CV. From K-fold CV, we used 10-fold CV by which 9 folds are used for training and the remaining one fold is used for testing iteratively. As the number of k values increases, the bias decreases in the model (Bruce G. , 2021). Therefore, we selected 10-fold CV rather than 5-fold CV.

## 3.8 Implementation Tools of Word Sense Disambiguation Model of Wolaita Language

### *3.8.1 Tools and Packages used*

This study used the python programing language for implementing and experimenting proposed solution. Python is used because of its programming choice for developers, researchers, and data scientists who need to work in machine learning models. Table 3.3 and Table 3.4 shows the list of tools and python packages with their version and description.

*Table 3. 4 Tools used for our study*

| ***Tools*** | ***Version*** | ***Description*** |
|---|---|---|
| **Anaconda navigator** | 3.7 | It is a python desktop graphical user interface used to provide conda packages, environments, and channels. |
| **Jupyter Notebook** | 3.7.4 | It is a python web application used to create live code, visualization, and narrative text. |
| **Spyder(Anaconda)** | 3.7.4 | The Spyder (Anaconda) is also used to write python code. |
| **Python** | 3.7 | easy to learn, powerful programming language to develop a machine learning application |
| **MathType** | 7.4.8.0 | Used to write mathematical formulas and symbols |
| **Microsoft Excel** | 2013 | Used data preparation tasks in cleaning filtering and sorting the data and used for annotation |
| **Microsoft Word** | 2013 | Used to write the documents |
| **Draw.io(online)** | | used to design the proposed solution of WSD model and prototype. |
| **SPSS Statistics data Editor** | 26 | The statistics software is used to calculate IRR of our data annotation |

In Table 3.4 below, we used python packages to implement the proposed WSD model and prototype of the Wolaita language. These packages were imported by python and perform a specific function. For example, the Scikit-learn package is used to import machine learning algorithms. The implementation of these packages was discussed in Section 5.

*Table 3. 5 List of Python Packages used in WSD*

| ***Python libraries used*** | ***Description*** |
|---|---|
| **Scikit-learn** | Python module that integrates classical machine learning algorithms |
| **NLTK** | The platform used to work with human language data. |
| **Pandas** | Used to read, write and manipulate data frames |
| **Numpy** | Used for process multi-dimensional array in python |
| **Flask** | It is a micro web framework used to develop web application |
| **Pickle** | Used to store and convert python objects in a database or file. |
| **Matplotib** | Used to visualize data in python |
| **Seaborn** | Used to visualize data in an attractive interface in python |
| **Re(Regular Expression)** | The re module allows us to search a string for a match in python. In this study, it is used for data cleaning and data normalization. |
| **String** | Byte of the array representing Unicode characters in python. |
| **Genism(Generate Similar)** | Python package used to build vectors and retrieve semantic similarity between documents |

### *3.8.2 Deployment Environment of Word Sense Disambiguation for Wolaita Language*

The tools which are discussed above in section 4.9 have been deployed on a personal computer equipped with a processor Intel(R) Core(TM) i5-4300M CPU @ 2.60GHz 2.60 GHz 4 Gigabyte of physical memory, 500 Gigabyte hard disk storage capacities. The operating system is Windows 10 Pro, 64-bit operating system, x64-based processor.

## 3.9 User interface Design

We designed a prototype of the WSD model using a python web framework called Flask. To design the user interface of WSD, we used Bootstrap that is an open-source framework for designing a responsive web application. The goal of designing a user interface is to test our model using new and unseen data.

# CHAPTER FOUR

## 4. PROPOSED WSD MODEL FOR WOLAITA LANGUAGE

### 4.1 The overview of proposed Solution of WSD for Wolaita Language

This chapter discusses the proposed solutions of WSD of Wolaita language using machine learning techniques. In this chapter, the proposed WSD for Wolaita language, proposed data preprocessing, proposed feature extraction, selected models, and model evaluation and testing are discussed.

### 4.2 Proposed Architecture of Word Sense Disambiguation Model for Wolaita Language

The proposed architecture of WSD for Wolaita language includes the data learning phase and classification phase. In the learning phase, the model first takes ambiguous sentences (sense examples) from the dataset and missed values set to zero using fillna (0) method in python. Then, these sense examples are tokenized to make the model to analyze the sequence of words. After performing tokenization, stop words are removed from the sense examples. And also to increase the quality of training data of the model symbols, numbers, punctuations, Emojis, and other irrelevant things are cleaned. In addition to these, some variations in the word forms are normalized into standard forms. Then these preprocessed data are transformed into the vector form. To convert text data into vector form, BOW, TF-IDF, and Word2Vec were used. The transformed data trained by the models such as SVM, Bagging, and Random Forest and AdaBoost classifiers. To increase the performance of these classifiers, parameters were tuned. These parameters are base estimators for Bagging Classifier and L2 regularization for all classifiers. The trained data is saved and dump into a pickle in the .sav file format. In the classification phase, the model predicts three and four classes (senses) using 10-fold CV by which nine folds are used for training and the rest one fold is used for testing iteratively. After that, the performance of the model was evaluated by confusion matrix, precision, recall, and f1-score.

In the classification phase, the model was evaluated by users by which users input new sentences with ambiguous words to the model and the model predicts the sense of meaning by comparing

new data with trained data that was saved in a pickle. If the probability of new data (confidence) is high, the model predicts the correct sense of meaning. Otherwise, the model tells the user to input the right data as depicted below in Figure 4.1

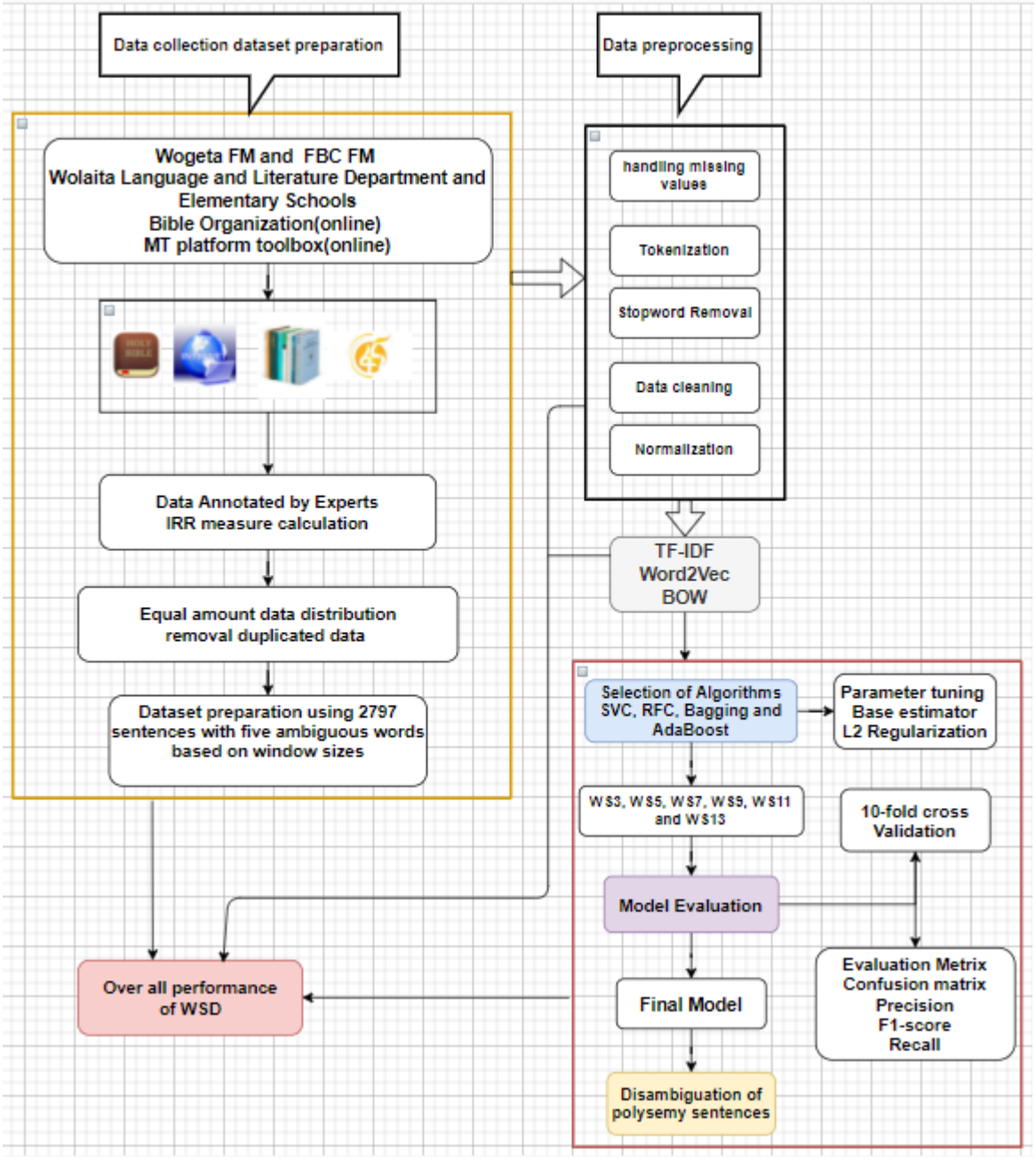



*Figure 4. 1 Proposed WSD Model of Wolaita Texts*

### 4.3 Proposed Data Preprocessing

Our data were collected from different kinds of sources as discussed in section 4.4. Data preprocessing play an important role for machine learning algorithms by providing suitable and clean data for the model to train easily. In this study, we proposed data preprocessing techniques such as tokenization, data cleaning, Stopword removal, and normalization.

***Handling missing data*:** As depicted in Appendix 4, the ambiguous words are put in the middle, and the rest words are put on the left and right sides of the ambiguous word. According to the morphology of the Wolaita language, words can appear in the beginning, middle, and end of sentences. Due to this, some column words didn't appear in the dataset. For example, 'Doona ubbaa issuwa oottiya eeshshay de'ees' meaning "There is characteristic that makes the language one'. In these sentence, the ambiguous word is "Doona' and appear at the beginning of the sentences. As a result, there are no left context words. Therefore, these are empty columns replaced by zero using python.

***Tokenization***: It splits sentences (sense examples) into the sequence of tokens that make a suitable environment for the model. Sense examples in our study were tokenized by WordTokenizer.

***Data cleaning*:** in this step punctuations, numbers, special characters, URLs, and Emojis were removed to train the model with quality data. These irrelevant things in our data are cleaned by the user-defined function called **Clean data ()** in python**.** as mentioned in the section**.**

***Stopword removal***: in these steps, unnecessary words that are not required for the model were removed. The Stopword of Wolaita language was collected from the previous researches and we put it into the Stopword list of NLTK folder in python. To remove these words, the **remove_stopword ()** function is defined in python. If the word in WSD data is a stop word, the function removes it.

***Normalization***: In this step, words that have the meaning but are spelled differently were changed into the standard form. As discussed in section 4.4, some words have the same meaning, but they are spelled differently and used interchangeably. Therefore, we normalized these words into standard form by **NormalizeData ()** function. If the variants of the words have the same

meaning, the function changes them to standard form. In figure 4.3 the data preprocessing steps are depicted.

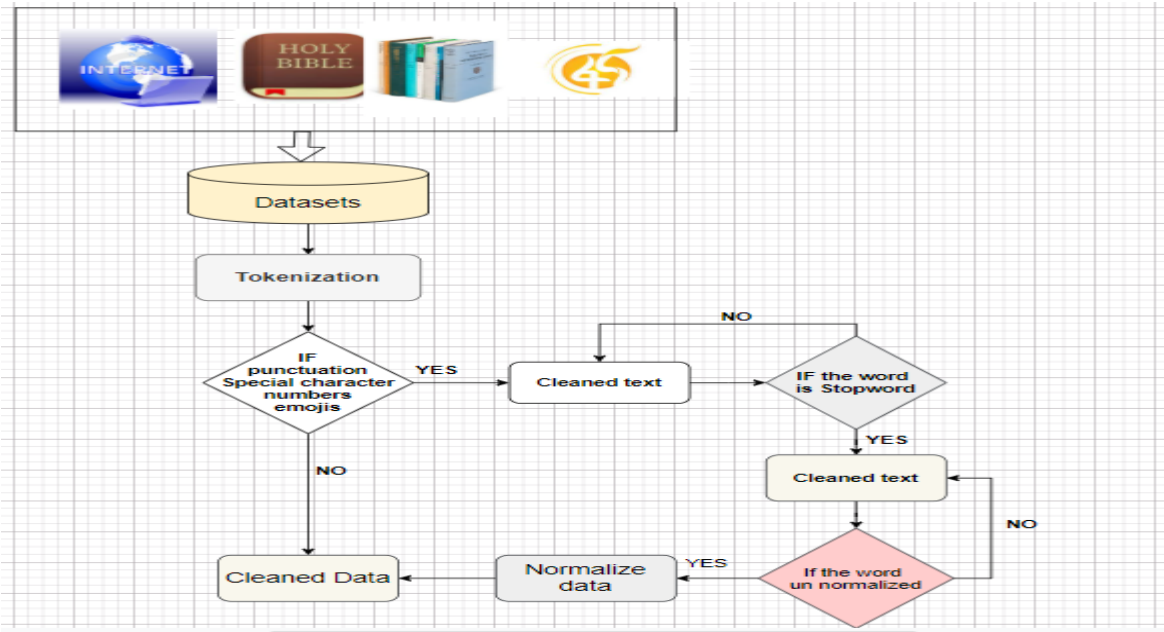



*Figure 4. 2 Data preprocessing Procedure*

### 4.4 Feature Extraction

The preprocessed (quality) data discussed above is not directly trained by the model. It is converted into a format that the model can understand. In this study, BOW, TF-IDF, and Word2Vec transformed data into vector form.

***Bag-of-Words:*** It is the measure of occurrence of words in the document. It is similar to CountVectorizer in Scikit Learn in python package. It uses Term Frequency to measure occurrence of term t in a document.

***Term Frequency (TF):*** It is the frequency measure of term **t** in a document. Some words that occur frequently in the document may have higher frequency values and the others may have lower frequency values. To handle these, the occurrence of the term **t** in a document is divided by the number of words in a document.

$$TF = \frac{count\ of\ t\ in\ document}{number\ of\ words\ in\ document} \quad \textbf{(4)}$$

After computing TF in Equation (4), the words are vectorized on Vocab (vocabulary of all possible words). But, in TF, stopwords (unnecessary words) have a higher frequency than others. To address, these problems IDF is used.

***Inverse Document Frequency (IDF):*** it measures the informativeness of term **t** in the document. Its measure is low for stopwords in the documents. During query time, some words may not find in the vocab and make df = 0. As numbers can't be divided by zero, we add 1 to the denominator as depicted below in Equation (5)

$$\boldsymbol{idf(t) = \frac{N}{df}} \quad \textbf{(5)}$$

Whereas df = occurrence of term t is calculated in Document N and Equation (5) calculates the document N over occurrence of term it. This shows that the measure of the frequent and unnecessary words have lowest value. Equation (6) used logarithmic function to handle the df = 0 in the above equation (6).

$$\boldsymbol{idf(t) = log \frac{N}{(df+1)}} \quad \textbf{(6)}$$

***TF-IDF***: is the combination of TF and IDF in which TF counts the frequency of term t in the document and IDF measures informativeness of term t in the document. Therefore, it is the product of two equations Equation (5) and Equation (6).

$$\boldsymbol{tf - idf(t,d) = tf(t,d) * log \frac{N}{(df+1)}} \quad \textbf{(7)}$$

Equation (7) shows the product of Equation (5) and Equation (6) to compute TF-IDF. Therefore, from the above equations, we ensure that TF-IDF feature extraction methods enhance the performance of models by giving lower values for the stop words and unnecessary frequent words.

***Word2Vec:*** It is one of neural network based feature extraction technique used mostly in NLP. In Word2Vec, the most similar word vector are chosen to represent semantic similarity between words. There are two model architectures used in Word2Vec to produce distributed representation of vocabulary of words. These are CBOW and Skip grams. We used COBW to predict the current word based on context window sizes of neighboring word. As depicted in figure 4.4, the similarity of word t achieved using the sum of context windows.

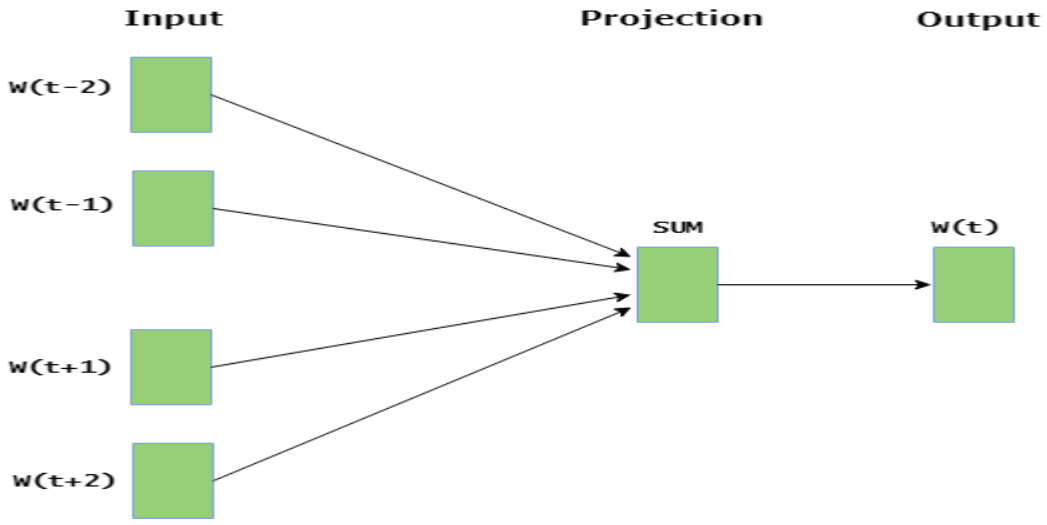



*Figure 4. 3 Representation of Wor2Vec using CBOW*

### 4.5 Machine Learning Models

In section 3.6, we selected SVC, Bagging, AdaBoost, and RFC for WSD of Wolaita language. The selected WSD Model of Wolaita language performs disambiguation of Wolaita sentences by first taking preprocessed and transformed data from the data preprocessing and feature extraction phase. Then classifiers learn from the train sets and evaluated by test sets using 10-fold CV. Finally, they accept the new data and predicts the meaning of the word according to the context. The models were trained based on six-window sizes (WS3, WS5, WS7, WS9, WS11, and WS13) to determine the better performance of our model. The result of six window sizes is mentioned in chapter six. For example, WS3 represents two words from the left side of the target word, two words from the right side of the target word, and the target word itself. The main goal of finding window size is to find the best window that increases the performance of the model.

In section 3.3, we discussed base estimators and regularization techniques. These techniques are parameters of the models and they are implemented with them. In our study, base estimators are implemented with Bagging Classifier and the L2 regularization technique is implemented with all selected models. We discussed each of them in section 5.4. The proposed models are depicted in figure 4.4

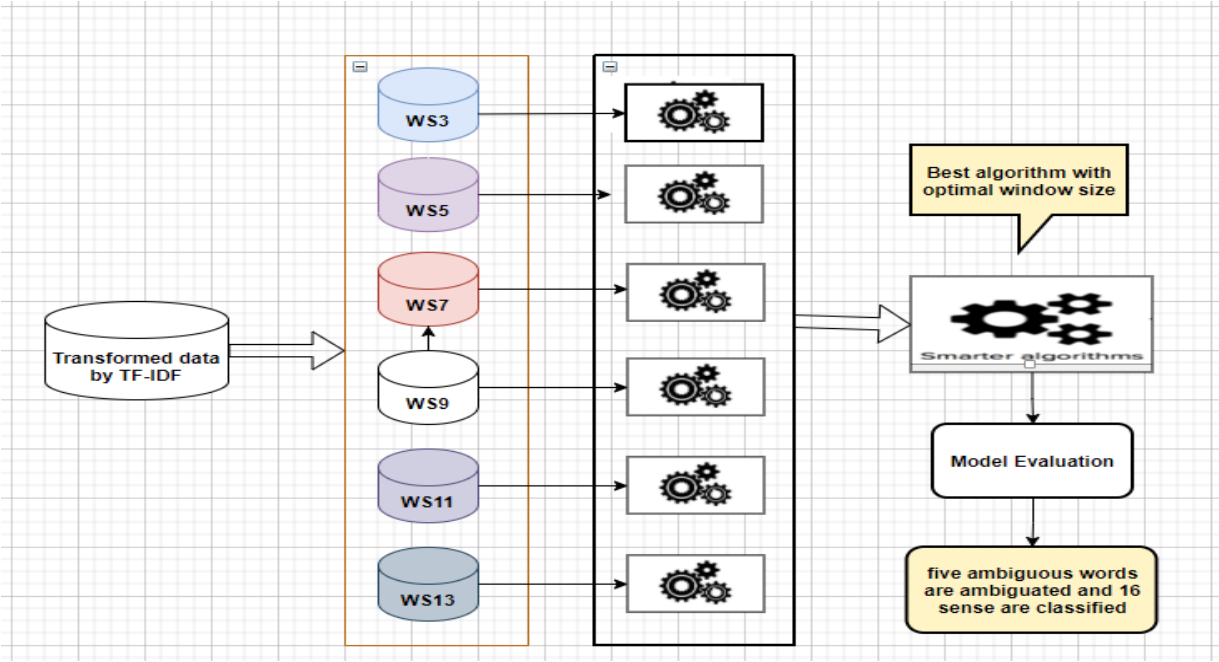



*Figure 4. 4 Selected WSD model of Wolaita Language*

### 4.6 Model Evaluation and Testing

As discussed in section 4.7, in the model evaluation and testing phase, k-fold CV and the model performance evaluation metrics (Confusion matrix, Accuracy, Precision, and Recall) are used. In 10-fold CV, 9 folds are used for training and the remaining one fold is used for testing iteratively. This iteration results accuracy for each fold and finally, their average accuracy is computed. The process of the K-Fold CV is explained as:

- ✓ Divide the data into K folds.
- ✓ Out of the K folds, K-1 sets are used for training whereas the remaining set is used for testing.
- ✓ The model is then trained and tested K times, each time a new set is used as testing whereas the remaining sets are used for training.
- ✓ Lastly, the result of the K-Fold CV is the average of the results obtained on each set.

***Confusion matrix***: It is N x N matrix used for model performance evaluation and it reports the number of false positives, false negatives, true positives, and true negatives by its rows and columns. It is used to compare the actual target values with the predicted value by the model.

***Accuracy***: in classification problems is the number of correct predictions made by the model over all kinds of predictions made as shown in Equation (8)

$$Accuracy = \frac{TF + TN}{TP + TN + FP + FN} \quad \textbf{(8)}$$

***Precision*:** It is the number of correct positive results divided by the number of positive results predicted by the classifier as shown in Equation (9)

$$Precision = \frac{TF + TN}{TP + FP} \quad \textbf{(9)}$$

***Recall*:** It is the number of correct positive results divided by the number of all relevant samples as shown in Equation (10)

$$\text{Recall} = \frac{TF + TN}{TP + FN} \quad \textbf{(10)}$$

***F1 score*:** It is the harmonic mean between precision and recall. It is used to measure the accuracy of tests and is a direct indication of the model's performance. The range of the F1 score is between 0 and 1 with the goal being to get as close as possible to 1. The formula of the F1-score is shown in Equation (11).

$$F1 = 2 * \frac{1}{\frac{1}{precision} + \frac{1}{recall}} \quad \textbf{(11)}$$

### 4.7 Proposed Prototype of Word Sense Disambiguation Model of Wolaita Language

In this phase, the model was deployed on a Flask web server. The model performance is evaluated by the users. First, the trained data was saved in .sav format and dumped into a pickle in the training phase. Then, this saved data was loaded on the Flask server and it responds to the request of the user through the WSGI server. This server takes the sentences with ambiguous words from users and it provides the sense of ambiguous words for users.

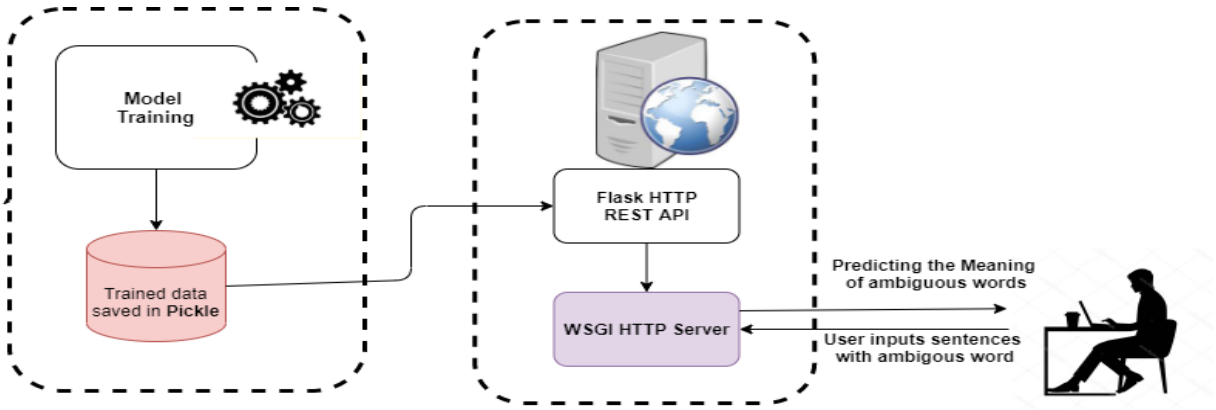



*Figure 4. 5 Prototype of WSD of Wolaita language*

# CHAPTER FIVE

# 5. IMPLEMENTATION OF PROPOSED WORD SENSE DISAMBIGUATION OF WOLAITA LANGUAGE

## 5.1 Introduction of Implementation and Experimentation

In this chapter, the implementation of the proposed WSD for Wolaita including the implementation of python packages used, the implementation of data preprocessing, the implementation of feature extraction, the implementation of models, and the implementation of model evaluation and testing techniques are discussed.

## 5.2 Data Preprocessing

### 5.2.1 Python packages and libraries

As discussed in section 3.8, we used python packages to implement the proposed solution of the WSD model of the Wolaita language. The packages used to implement WSD are depicted below in figure 5.2. After implementing the python packages, we load the dataset using the **pandas** package. To handle missing data in the dataset we used **fillna** (0) method to fill zero in empty cells in the dataset as depicted in figure 5.1

```
import pandas as pd
df = pd.read_csv(r'wsd_dataset.csv', header = None,encoding='latin-1',)
df=df.fillna(0)
df.head(5)
```

| LeftContext_5 | LeftContext_4 | LeftContext_3 | LeftContext_2 | LeftContext_1 | Target_word | RightContext_1 | RightContext_2 | RightContext_3 | RightContext_4 | Righ |
|---|---|---|---|---|---|---|---|---|---|---|
| Toophphiya | luxetta | yunbburshshetun | timirtte | luxetta | doona | gidiis | 0 | 0 | 0 | |
| 0 | 0 | Olaappe | simmin | Inggilizetto | doonay | pirdda | keettatuuninne | kawo | ooso | |
| amarida | Toophphiya | xaafanchchati | Inggilizettuwa | naa□antto | doonadan | ekkidosona | 0 | 0 | 0 | |
| 0 | 0 | 0 | 0 | Inggilizetto | doonay | Toophphiya | xaafanchchatun | doorettido | gaasoy | xaafan |
| ha | aahuwan | Luma | Gezuwan | goggiya | doonan | aaho | deriyakko | gakkanawu | ogiya | d |

*Figure 5. 1 Implementation of Data loading and handling missing values*

```
import nltk
import re
import string
import pickle
import pandas as pd
import numpy as np
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.pipeline import Pipeline
# algorithms used
from sklearn.svm import LinearSVC
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import AdaBoostClassifier
#Base estimators used in ensemble classifiers(RFC,Bagging and Adaboost)
from sklearn.linear_model import SGDClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
#packages used to evaluate our models
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import cross_val_predict
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report, confusion_matrix
# packages used to visualize data
import seaborn as sns; sns.set()
import matplotlib.pyplot as plt
%matplotlib inline
```

*Figure 5. 2 Python Packages and Libraries used*

### *5.2.2 Tokenization*

In NLP, Tokenization is the building block of Models. It is used to analyze the sequence of words by which the meaning of the text is interpreted. As the Wolaita language is Latin script, we tokenized the sentences as English. We imported word_tokenize from NLTK and tokenized the datasets as depicted in figure 5.3.

```
from nltk.tokenize import  word_tokenize

df['sense_example1'] = df.apply(lambda row: nltk.word_tokenize(row['sense_example1']), axis=1)
df['sense_example2'] = df.apply(lambda row: nltk.word_tokenize(row['sense_example2']), axis=1)
df['sense_example3'] = df.apply(lambda row: nltk.word_tokenize(row['sense_example3']), axis=1)
df['sense_example4'] = df.apply(lambda row: nltk.word_tokenize(row['sense_example4']), axis=1)
df['sense_example5'] = df.apply(lambda row: nltk.word_tokenize(row['sense_example5']), axis=1)
```

*Figure 5. 3 Tokenization of WSD datasets*

### *5.2.3 Stopword Removal*

Stopwords are commonly used words in human language. But in the NLP, they are not required to train the model. Because we need important features to train the model. To remove Wolaita stopwords, we imported NLTK packages as discussed above. From the NLTK package, we imported the stopwords removal library and we removed stopwords using the remove_stopword () function as depicted in figure 5.4.

```
STOPWORDS = set(stopwords.words('wolaita'))
def remove_stopwords(text):
    """custom function to remove the stopwords"""
    return " ".join([word for word in str(text).split() if word not in STOPWORDS])

df["sense_example1"] = df["sense_example1"].apply(lambda text: remove_stopwords(text))
df["sense_example2"] = df["sense_example2"].apply(lambda text: remove_stopwords(text))
df["sense_example3"] = df["sense_example3"].apply(lambda text: remove_stopwords(text))
df["sense_example4"] = df["sense_example4"].apply(lambda text: remove_stopwords(text))
df["sense_example5"] = df["sense_example5"].apply(lambda text: remove_stopwords(text))
```

*Figure 5. 4 Stopword removal of WSD data*

### *5.2.4 Data Cleaning*

As discussed in section 3.3, we collected the data from different sources. These collected data contain numbers, punctuations, special characters, and also Emojis. These irrelevant things should be cleaned to provide quality data for the model. To clean these we used the **clean_data ()** function and **Regular Expression** (re) to replace irrelevant things with space as depicted in figure 5.5 below.

```
def clean_data(text_string):
    space_pattern = '\s+'
    url_regex = ('http[s]?://(?:|[0-9]|[$-_@.&+]|''[!*\(\),]|(?:%[0-9a-fA-F][0-9a-fA-F]))+')
    special_char = "[@#$%^&=?x!,;:_.(){}`'/+*<>\"¤– „\ ®˜™¡¡\x10»€«·‘01§”¬¦…""f÷\~¨©±¥£¶-°•˜’“|]"
    RE_EMOJI = re.compile('[\U00010000-\U0010ffff]', flags=re.UNICODE)
    Clean_text = re.sub(url_regex,'', text_string)
    Clean_text=re.sub(special_char,' ',Clean_text)
    #Clean_text= re.sub(r'(.)\1+', r'\1\1',Clean_text) #removing elongation in text
    Clean_text = RE_EMOJI.sub(r'', Clean_text)
    Clean_text=re.sub('-','',Clean_text)
    Clean_text=re.sub(r'<[^>]*>','', Clean_text)
    Clean_text=re.sub('-',' ',Clean_text)
    Clean_text = Clean_text.replace("\\", "");
    Clean_text = Clean_text.replace("[", "");
    Clean_text = Clean_text.replace("]", "");
    Clean_text = re.sub(space_pattern,' ',Clean_text)
    return Clean_text

df["sense_example1"] = df["sense_example1"].apply(lambda text: clean_data(text))
df["sense_example2"] = df["sense_example2"].apply(lambda text: clean_data(text))
df["sense_example3"] = df["sense_example3"].apply(lambda text: clean_data(text))
df["sense_example4"] = df["sense_example4"].apply(lambda text: clean_data(text))
df["sense_example5"] = df["sense_example5"].apply(lambda text: clean_data(text))
```

*Figure 5. 5 Data Cleaning*

### *5.2.5 Normalization*

Then after cleaning data in the above section, we normalized words that are similar in meaning but spelled differently. In addition to these, we normalized the letters such as (ai and oi) used in Holy Bible into the standard form (ay and oy). The words and letters to be normalized are listed in the dictionary and the user-defined function **Normalize** () checks the words with the dictionary. If the matches are found in the dictionary, it normalizes into the standard form. The implementation of normalization is depicted in figure 5.6

```
# Dictionary of words to be norimalized
word_dic = { "ai": "ay","ai": "ay","oi": "oy","au": "awu"," diya": "de'iya"," daana": "de'ana",
                    "alamiya":"salo-gufantto "," katamaa ": "allaana ","geetettiya ": "giyo","uri ": "asi",
                    "yihoowaa ": "Xoossaa","Dr.": "Dotore","Prof. ": "Pirofeesere",}

# Regular expression for finding contractions
word_re=re.compile('(%s)' % '|'.join(word_dic.keys()))

# Function for expanding contractions
def normalize(text,word_dic=word_dic):
    def replace(match):
        return word_dic[match.group(0)]
    return word_re.sub(replace, text)

# Expanding Contractions in the title, text
df['sense_example1'] = df['sense_example1'].apply(lambda text: normalize(text))
df["sense_example2"] = df["sense_example2"].apply(lambda text: normalize(text))
df["sense_example3"] = df["sense_example3"].apply(lambda text: normalize(text))
df["sense_example4"] = df["sense_example4"].apply(lambda text: normalize(text))
df["sense_example5"] = df["sense_example5"].apply(lambda text: normalize(text))
```

*Figure 5. 6 Normalization of Data*

### 5.3 Feature Extraction

The cleaned data in the above section was transformed into vector form using three feature extraction techniques. These are BOW, TF-IDF and Word2Vec. The implementation of each of them shown below.

#### *5.3.1 Bag of Word using CountVectorizer*

In this study, we used CountVectorizer from Scikit learn to implement Bag of Word. We used pipeline to vectorize the text data and to train the model. To implement CountVectorizer, we used the **fit ()** method in python. The implementation of CountVectorizer is depicted in figure 5.7

```
%%time
bag = Pipeline([('vect', CountVectorizer(ngram_range=(1, 3))),
                ('clf',  AdaBoostClassifier(n_estimators =100))])
bag.fit(WS9, y)
```

*Figure 5. 7 Implementation of CountVectorizer*

### *5.3.1 TF-IDF*

In this study, we used TfidfTransformer in the pipeline to vectorize the text data to be trained by the model. To implement TF-IDF, we used CountVectorizer as TF and IDF that ignores the irrelevant features from TF. To transform data, we used the **fit ()** method in python. The implementation of TF-IDF is depicted in figure 5.8

```
%%time
bag = Pipeline([('vect', CountVectorizer(ngram_range=(1, 3))),
                ('tfidf', TfidfTransformer()),
                 ('clf',  BaggingClassifier(n_estimators =100, base_estimator=clf3))])
bag.fit(WS9, y)
```

*Figure 5. 8 Feature extraction using TF-IDF*

### *5.3.2 Word2Vec*

In this study, we used Word2Vec to vectorize the text data and to compute semantic similarity between vectors of words. To implement Word2Vec, we used genism package from python. The implementation of Word2Vec is depicted in figure 5.9.

```
model = gensim.models.Word2Vec(x_tokenized,
                               window=3
                    )
model.wv.most_similar("daro")
```

*Figure 5. 9 Implementation of Word2Vec*

## 5.4 Selected Models of Word Sense Disambiguation

The implementation of machine learning models for WSD for Wolaita language was implemented in three experiments regarding window sizes. As discussed in section 4.5, we selected six window sizes (WS3, WS5, WS7, W9, WS11, and WS13).

We tuned the algorithms' parameters to implement the WSD model of the Wolaita language. Among these, we are interested to investigate the effect of base estimators in the Ensemble classifiers. In addition to these, we implemented regularization techniques to handle the problem of overfitting. The regularization technique used in this study was L2 regularization. The implementation of four classifiers such as Bagging, SVM, Random Forest, and AdaBoost classifiers.

After integrating the methods discussed above, we trained the model using the **fit ()** method. Finally, the trained data is converted into a byte stream in .sav format using the **pickle** module in python. Later this data is loaded to predict the new user input in the classification phase. The implementation of SVC, RFC, Bagging Classifier, and AdaBoost Classifier is depicted below.

### *5.4.1 Base Estimators for Bagging Classifier*

In ensemble methods, the predictions of base estimators are combined to make generalizability of the algorithm. In another word, several weak models make the algorithm powerful. By default, ensemble methods use a Decision Tree classifier. To know the effect of other base estimators such as Linear SVC, SGD, and Logistic regression, we implemented base estimators with our selected models especially Bagging and AdaBoost Classifier.

In base estimator implementation, we used L2 regularization to handle the problem of overfitting using **penalty** parameters. We set the class weight balanced to handle the problem of class imbalance in the model. The implementation of base estimators depicted in figure 5.10

```
# The implementation of Base_estimator or Weak Learner in Ensemble Methods
clf1 =clf = DecisionTreeClassifier(class_weight='balanced',max_depth=100)
clf2 = LinearSVC(penalty = 'l2',class_weight='balanced', C = 100, dual = False)
clf3 = SGDClassifier(loss="hinge", max_iter=200,penalty="l2")
clf4= LogisticRegression(n_jobs=1, multi_class='ovr',class_weight='balanced',solver='liblinear',C=1e5)
```

*Figure 5. 10 Implementation of Base estimators*

### *5.4.2 Bagging Classifier*

As discussed in the above section, we implemented a Bagging classifier with four base estimators. The parameters used in the base estimators are also used in the Bagging algorithm. The trained data is saved in the pickle database as discussed above. The effect of base estimators is described in chapter seven (result and discussion).

```
bag = Pipeline([('clf',  BaggingClassifier(n_estimators =100, base_estimator=clf1))])
bag.fit(x_conv, y)
filename = 'bag.sav'
pickle.dump(bag, open(filename, 'wb'))
```

*Figure 5. 11 Implementation of Bagging Classifier*

### *5.4.3 AdaBoost Classifier*

Similarly, we implemented the AdaBoost classifier with four base estimators. The parameters used in the base estimators are also used in the AdaBoost algorithm. The trained data is saved in the pickle database as discussed above. The effect of base estimators is described in chapter seven (result and discussion).

```
adb = Pipeline([('clf',  AdaBoostClassifier(n_estimators =100,base_estimator=clf4))])
adb.fit(x_conv, y)
filename = 'adb.sav'
pickle.dump(adb, open(filename, 'wb'))
```

*Figure 5. 12 The implementation of AdaBoost Classifier*

### *5.4.4 Random Forest Classifier*

We implemented RFC with the parameters such as class weight balanced. Unlike the above classifiers, this classifier uses a built-in L2 regularization and Decision Tree base estimator. Therefore, to increase the performance of the model, Decision Trees are tuned with the number of estimators. Then, the trained data is saved in the **pickle** database as discussed above.

```
rfc = Pipeline([('clf',  RandomForestClassifier(n_estimators =100, min_samples_split=3,class_weight='balanced'))
                ])
rfc.fit(x_conv, y)
filename = 'rfc.sav'
pickle.dump(rfc, open(filename, 'wb'))
```

*Figure 5. 13 The Implementation of Random Forest Classifier*

### *5.4.5 Support Vector Classifier*

Unlike ensemble methods, SVC uses a hyperplane to classify the data points. In our study, we used a linear kernel to classify the sense examples. To handle the problem of overfitting, we used L2 regularization. In addition to these, we made the class weight balanced to avoid the occurrence of an imbalance class. Finally, we trained the model using the **fit ()** method, and the trained data was converted into a byte stream in .sav format using the **pickle** module in python.

```
svm = Pipeline([('clf',  LinearSVC(penalty = 'l2',class_weight='balanced', C = 100, dual = False)),
                ])
svm.fit(x_conv, y)
filename = 'svm.sav'
pickle.dump(svm, open(filename, 'wb'))
```

*Figure 5. 14 The Implementation of SVC*

### 5.5 Model Evaluation Techniques

We evaluated our models by using 10-fold CV. From the 10 folds, 9 folds are used for training and 1 fold for testing iteratively. In figure below **cross_val_score ()** and **cross_val_predict ()** methods are used to evaluate our models. Cross_val_score is used to calculate the average accuracy of the 10 folds while cross_val_predict is used to return the predicted label. In addition to this, we evaluated our model performance using evaluation metrics such as confusion matrix, precision, recall, and F1-score. The implementation of model evaluations depicted in figure 5.15

```
CVscore=cross_val_score(svm, x_conv, y, cv=10)
y_pred = cross_val_predict(svm, x_conv, y, cv=10)
CVscore.mean()
print(CVscore.mean())
print('accuracy %s' % accuracy_score(y_pred, y))
print(classification_report(y, y_pred))
```

```
conf_mat = confusion_matrix(y,y_pred)
sns.heatmap(conf_mat.T, square=True, annot=True, fmt='d', cbar=False,
            xticklabels=np.unique(y),
            yticklabels=np.unique(y))
plt.xlabel('True Entity')
plt.ylabel('Predicted Entity')
```

*Figure 6. 15 The Implementation of Model Evaluations*

### 5.6 Prototype Design of Word Sense Disambiguation Model

As discussed in section 3.8, we used the Flask web server in python and the bootstrap to design the prototype of WSD of the Wolaita language. As mentioned in Appendix 8, the data is saved in a pickle and if the sentences from the user have the highest probability when compared with pickle data, the model predicts the sense of ambiguous words.

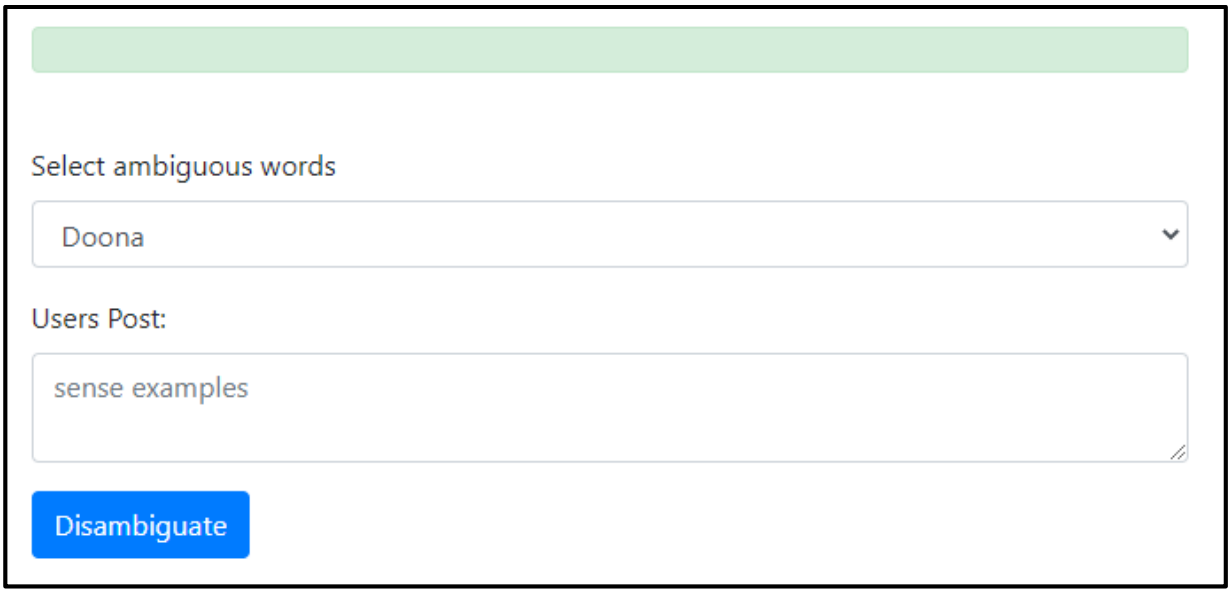



*Figure 5. 16 The Implementation of Model Prototype*

# CHAPTER SIX

# 6. RESULT AND DISCUSSIONS

## 6.1 Introduction

In this chapter, the result of the experiments of the proposed solution for WSD for Wolaita language using machine learning is discussed. Here, the experimental result of feature extraction techniques, the result of model evaluation, and the experimental result of context window sizes on the model performance are discussed. In addition to these, we discussed the effect of base estimators in the Bagging Classifier. Finally, we discussed the significance and the findings of the result obtained by each experiment in this study.

## 6.2. Selection of Feature Extraction Techniques based on Average accuracy

As discussed in section 3.5, three feature extraction techniques are selected in our study to transform the text data into vector form. To select the best feature extraction techniques, we conducted the experiment on BOW, TF-IDF, and Word2Vec.The experimental result of each of them listed below in Table 6.1.

*Table 6. 1 The Experimental Result of Feature Extraction Techniques applied for WSD*

| | ***Experimental Result of Feature Extraction using Four Classifiers*** | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ***SVM*** | | | ***Bagging*** | | | ***RFC*** | | | ***AdaBoost*** | | |
| **Feature Extraction Techniques** | ***BOW*** | ***TF-IDF*** | ***Word2Vec*** | ***BOW*** | ***TF-IDF*** | ***Word2Vec*** | ***BOW*** | ***TF-IDF*** | ***Word2Vec*** | ***BOW*** | ***TF-IDF*** | ***Word2Vec*** |
| Doona | 83 | 85.3 | 71.3 | 85 | 85.1 | 76.6 | 80.8 | 82.8 | 74 | 70 | 69.2 | 69 |
| Ayfiya | 79 | 83.6 | 66.7 | 80.3 | 83.7 | 70 | 75 | 73.6 | 78 | 72.5 | 69.3 | 63.5 |
| Ogiya | 76.2 | 83.6 | 67 | 78.8 | 84.0 | 67.8 | 72 | 78.2 | 67.2 | 69.6 | 67.6 | 63 |
| Naaga | 77.5 | 82 | 53.4 | 80.3 | 82.1 | 63.7 | 73.5 | 78.3 | 68.3 | 54.7 | 74.3 | 54.5 |
| Aadhdha | 81.8 | 81.6 | 68.6 | 81.8 | 80.5 | 70 | 79.6 | 73.5 | 70 | 71.9 | 71.3 | 67.3 |
| Average | 79.5 | 83.22 | 65.4 | 81.24 | 83.08 | 69.62 | 76.18 | 77.28 | 71.5 | 67.74 | 70.34 | 63.46 |

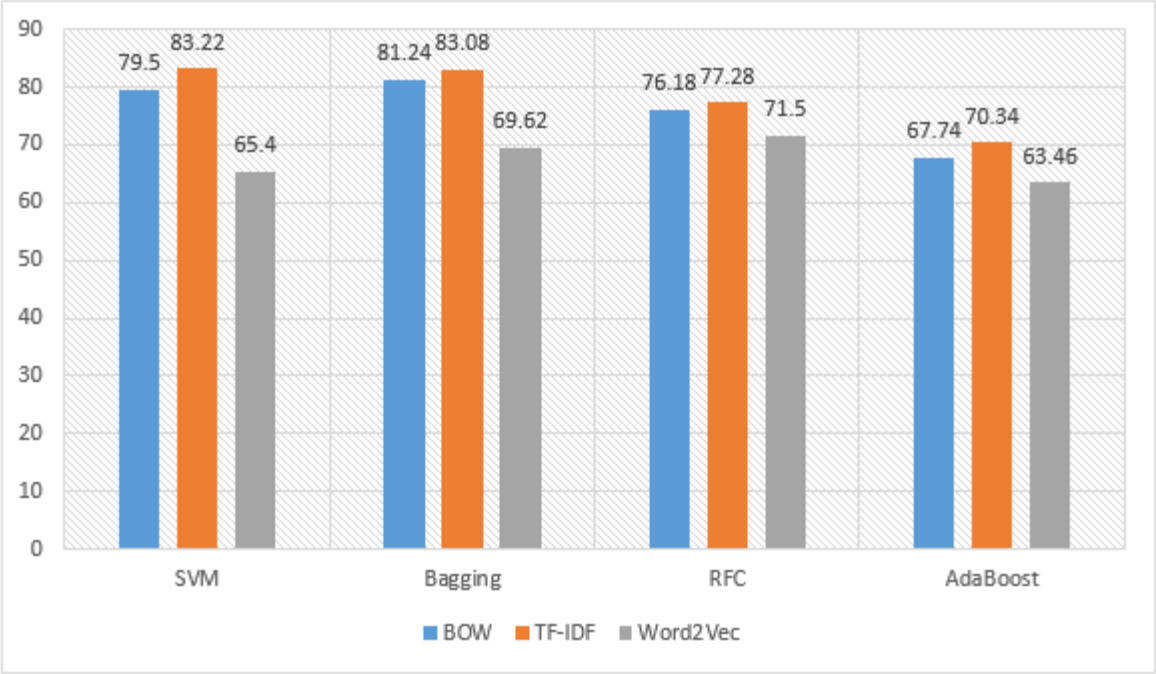



*Figure 6. 1 Experimental Result of Feature Extraction Techniques used in WSD*

As the experimental result of Table 6.1 and Figure 6.1 shows, the TF-IDF transformer achieved greater accuracy than BOW and Word2vec. As we discussed in section 3.5, TF-IDF works best for the larger and smaller datasets and also it ignores the irrelevant and frequent words and gives the significance to the important features. This is again seen in this experiment. Therefore, we conclude that TF-IDF is best transformer for WSD model of Wolaita language. The experiments below also conducted based on TF-IDF.

### 6.3 The Experimental Result of Base Estimator (Weak Learners)

As discussed in 5.4, ensemble methods[6] use weak learner algorithms, to sum up their average weight when training the data. In this study, we selected Bagging Classifier from ensemble methods. By default ensemble methods use DT as a weak leaner. In this study, we experimented on other weak learners such as Linear SVC, Logistic regression, and SGD algorithms in terms of performance (accuracy). We made an experiment on these base learners using five datasets. The experimental result (CV result) of these base estimators (weak learners) is listed below.

[6] https://scikit-learn.org/stable/modules/ensemble.html

*Table 6. 2 Experimental Result of Bagging Classifier using five Dataset*

| Weak learners | ***Aadhdha*** | ***Doona*** | ***Ayfiya*** | ***Naaga*** | ***Ogiya*** | ***Average*** |
|---|---|---|---|---|---|---|
| DT | 76 | 74.1 | 74.7 | 76 | 74.1 | 74.98 |
| Linear SVC | 85.1 | 83.7 | 84 | 80.1 | 81.2 | **82.82** |
| Logistic Regression | 84.1 | 83.7 | 84 | 80.1 | 81.2 | 82.62 |
| SGD | 81.4 | 84.1 | 82 | 82.4 | 83.5 | 82.68 |

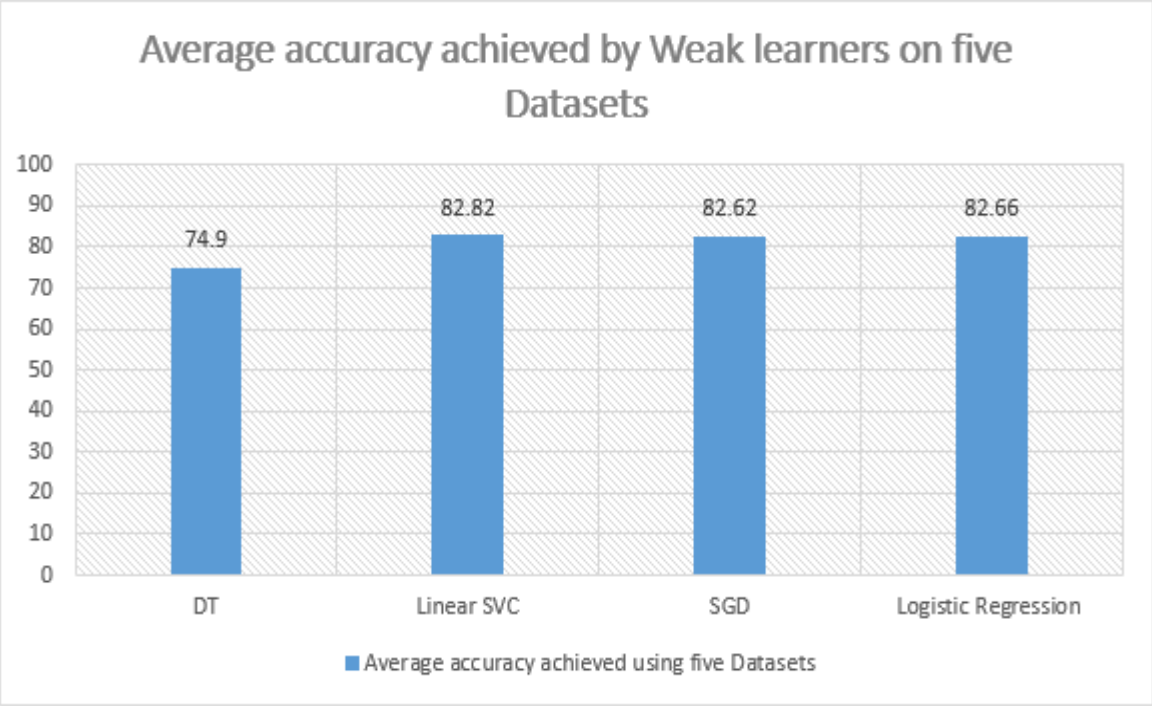



*Figure 6. 2 Experimental Result of Weak learners of Bagging Classifier*

As Table 6.2 and Figure 6.2 show that the Bagging Classifier with Linear SVC weak learner achieved higher result than other estimators in terms of performance. The default base estimator called DT achieved low accuracy than other weak learner algorithms. We conducted this experiment to select the best multi-class classification weak learners that can increase the performance of the Bagging classifier that was selected for our model. As we discussed in section 5.4, bagging classifier has the parameter called base estimator that supports weak learner algorithms. Instead of using default base estimator DT, it is better to select other base estimators

to enhance the performance of the classifier. Therefore, we concluded that Bagging Classifier with Linear SVC base estimator (weak learner) is better for WSD model Wolaita language.

The other ensemble classifiers such as RF and AdaBoost don't support the base estimators discussed above. Therefore we took default base estimators for these classifiers. On the other hand, SVC is different from ensemble methods. It uses a hyperplane to classify data points rather than using weak learners (base estimators). In the following experiments, we conducted the effect of context window size on the five datasets using Bagging with base estimator Linear SVC, SVC, Random Forest, and AdaBoost Classifiers.

### 6.4 The selection of optimal Context Window Size

In this study, we experimented on window size to find the optimal window size of the WSD model of Wolaita Language. As discussed in section 2.8(related works), many types of research were conducted to find optimal window sizes for many languages. Therefore we experimented on window size on five datasets (Doona, Ayfiya, Aadhdha, Naaga, and Ogiya) to find optimal window size using four Algorithms discussed above. As stated by (WELEMICHAEL, 2018), the optimal window size for datasets is obtained by calculating the average accuracy achieved in each window size. The accuracy achieved in each window and the average accuracy of whole window sizes are shown in Table 6.3.

*Table 6. 3 The Experimentation Result of Window size using four Classifiers*

| | | ***Five Datasets*** | | | | | |
|---|---|---|---|---|---|---|---|
| ***Classifiers*** | ***Window Sizes(WS)*** | ***Doona*** | ***Ayfiya*** | ***Aadhdha*** | ***Ogiya*** | ***Naaga*** | ***Average*** |
| **Bagging Classifier** | WS3(1-1) | 82.6 | 81.2 | 80.6 | 77.8 | 79.5 | 80.18 |
| | WS5(2-2) | 83.9 | 79.6 | 85 | 79.5 | 80.1 | 80.84 |
| | WS7(3-3) | 82.4 | 81.7 | 83 | 77.9 | 81.1 | 80.1 |
| | WS9(4-4) | 86.1 | 82.66 | 82.9 | 82.8 | 81.1 | 82.952 |
| | WS11(5-5) | 85.1 | 83.7 | 84.0 | 80.1 | 81.2 | **82.82** |
| | WS13(6-6) | 84.7 | 84.6 | 83.6 | 78.3 | 81.6 | 82.6 |
| **Support Vector Classifier** | WS3(1-1) | 82.2 | 81.4 | 85.0 | 77 | 79.7 | 81.06 |
| | WS5(2-2) | 84.6 | 79.1 | 84.0 | 79 | 80.2 | 81.38 |
| | WS7(3-3) | 82.2 | 80.9 | 84.0 | 77.5 | 82.5 | 81.42 |
| | WS9(4-4) | 85.4 | 82.37 | 83.4 | 82.2 | 80.9 | 82.854 |
| | WS11(5-5) | 85.3 | 83.6 | 83.6 | 82 | 81.6 | **83.22** |
| | WS13(6-6) | 84.16 | 83.7 | 83.2 | 78.2 | 82.1 | 82.272 |
| **Random Forest Classifier** | WS3(1-1) | 77.8 | 71.5 | 74.5 | 75.2 | 75.0 | 74.8 |
| | WS5(2-2) | 79.5 | 71.7 | 79.2 | 77.4 | 72.8 | 76.12 |
| | WS7(3-3) | 77.9 | 72.8 | 78.6 | 77 | 74.7 | 76.2 |
| | WS9(4-4) | 82.8 | 73.6 | 78.2 | 78.3 | 73.5 | **77.28** |
| | WS11(5-5) | 82.1 | 75.2 | 77.9 | 78.2 | 71.6 | 77 |
| | WS13(6-6) | 74.8 | 75.3 | 77.4 | 74.3 | 72.6 | 74.88 |
| **AdaBoost** | WS3(1-1) | 69.2 | 69.3 | 71.3 | 67.6 | 74.3 | **70.34** |
| | WS5(2-2) | 70.7 | 67.5 | 73.0 | 60.2 | 54.2 | 65.12 |
| | WS7(3-3) | 69.6 | 65.1 | 72.3 | 65.8 | 58.0 | 66.16 |
| | WS9(4-4) | 72.2 | 69.7 | 74.3 | 67 | 53.4 | 67.32 |
| | WS11(5-5) | 68.9 | 68.2 | 73.1 | 67.3 | 55.5 | 66.6 |
| | WS13(6-6) | 68.2 | 66.5 | 74.9 | 67.6 | 55.1 | 66.46 |

As can be seen from Table 6.3, the highest accuracy achieved by SVC, Bagging Classifier on WS11 (5-5). The average accuracy achieved by these classifiers are **83.22**%, and **82.82**% respectively. In addition to, this RFC achieved **77.28**% on WS9 (4-4) and AdaBoost Classifier achieved **70.34** % on WS3 (1-1). Based the accuracy achieved by SVC and Bagging Classifier, WS11 (5-5) selected as the optimal window size of WSD model of Wolaita language. We also experimented to know the effect of large window that makes computational time worst. As claimed by (Daniel Jurafsky & James H. Martin, 2006), the larger window makes the computational time worst for the classifiers. And also smaller window sizes make syntactic meaning rather than semantic. In our study, we selected the neither smaller window size nor larger window sizes.

*Table 6. 4 Experiment on Window size in terms of computational cost using SVC*

| ***Window sizes*** | ***Time taken(sec) for five datasets using SVC*** | | | | | |
|---|---|---|---|---|---|---|
| | ***Doona*** | ***Ayfiya*** | ***aadhdha*** | ***naaga*** | ***ogiya*** | ***average*** |
| WS3 | 3.21 | 4.64 | 3.18 | 4.4 | 4.01 | 3.888 |
| WS5 | 5.38 | 6.14 | 4.58 | 10.6 | 6.49 | 6.638 |
| WS7 | 5.84 | 7.92 | 6.34 | 8.42 | 8.76 | 7.456 |
| WS9 | 7.07 | 9.8 | 6.72 | 8.6 | 8.72 | 8.182 |
| WS11 | 7.92 | 9.81 | 9.81 | 11 | 12.6 | 10.228 |
| WS13 | 8.24 | **10.4** | 8.29 | **11** | 16 | 10.786 |

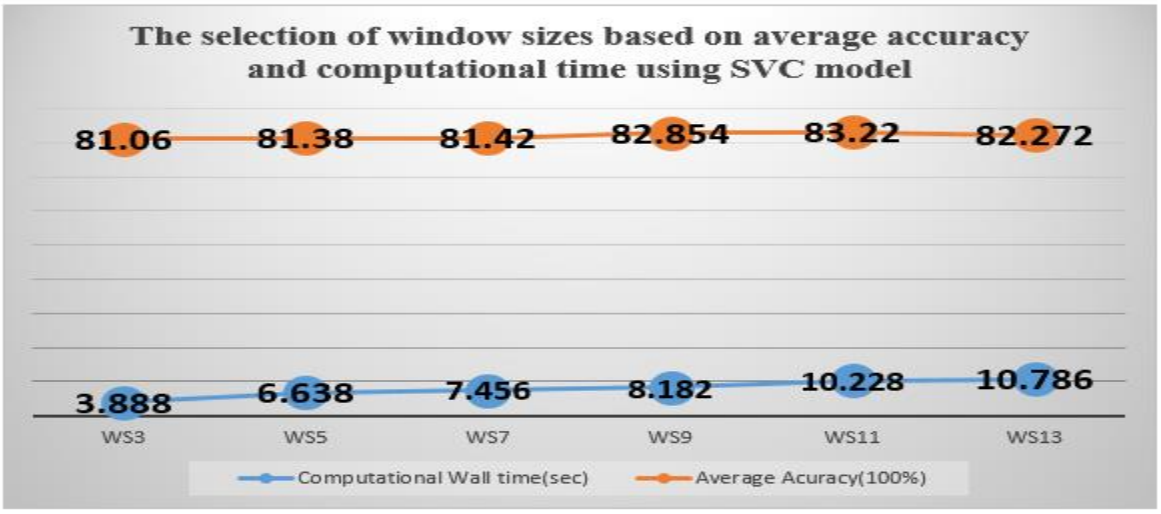



*Figure 6. 3 Experimental results of window size in terms of Computational Cost*

As shown on figure 6.3 the computational time of SVC increases with increasing window sizes. This indicates that the largest window sizes are worst in terms of computational time if the number of data increases. As stated by (GEOFFREY I, 2001), many models were developed to assist the user requirement through the internet. The amount of data processed and the number of users has been increasing. Therefore considering the models in terms of the computational time increases the viability of the model. Based on these, we selected WS11 (5-5) for WSD of Wolaita language.

### 6.5 Model Evaluation Result

In this section, we evaluated the models based on the optimal window size that was discussed in the above section. Instead of using all window sizes, it is better to use optimal window sizes that enhance the model in terms of performance and the computational time it costs. We evaluated our model using two evaluation techniques. These are the model performance evaluation metrics (Confusion matrix, Accuracy, Precision, Recall, and F1-score) and human evaluation by which we evaluated our model by unseen data. In the following section, we evaluated two models (SVC and Bagging Classifier) that achieved higher accuracy in the above section.

### *6.5.1 Confusion Matrix Result of Support Vector Machine using Five Datasets*

In this section, we discussed the result of the confusion matrix using five datasets (Doona, Ayfiya, Aadhdha, Ogiya, and Naaga) based on WS11 (5-5).

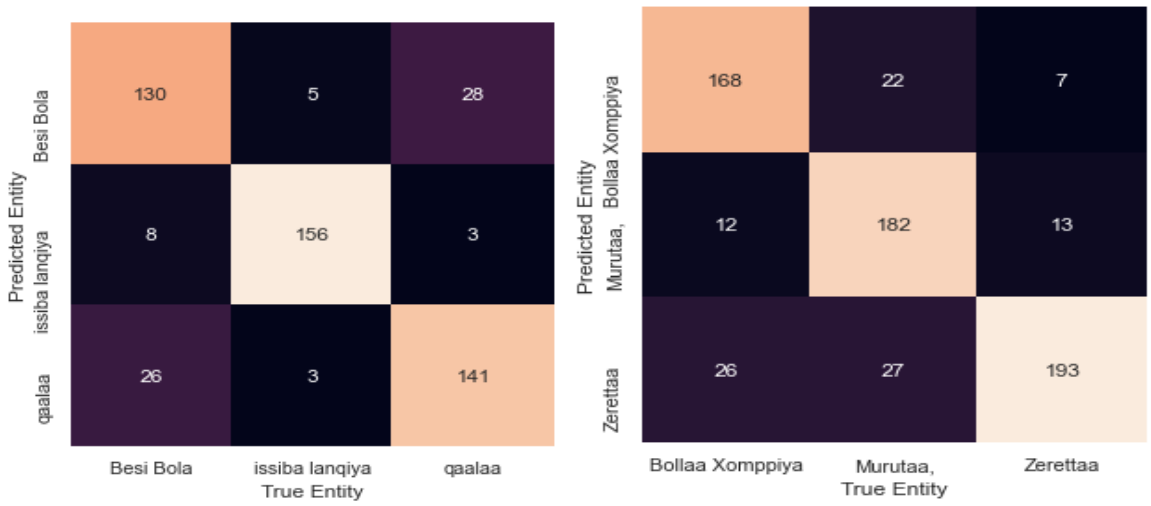



*Figure 6. 4 Result of SVC Confusion matrix using Left (Doona dataset) and right (Ayfiya dataset)*

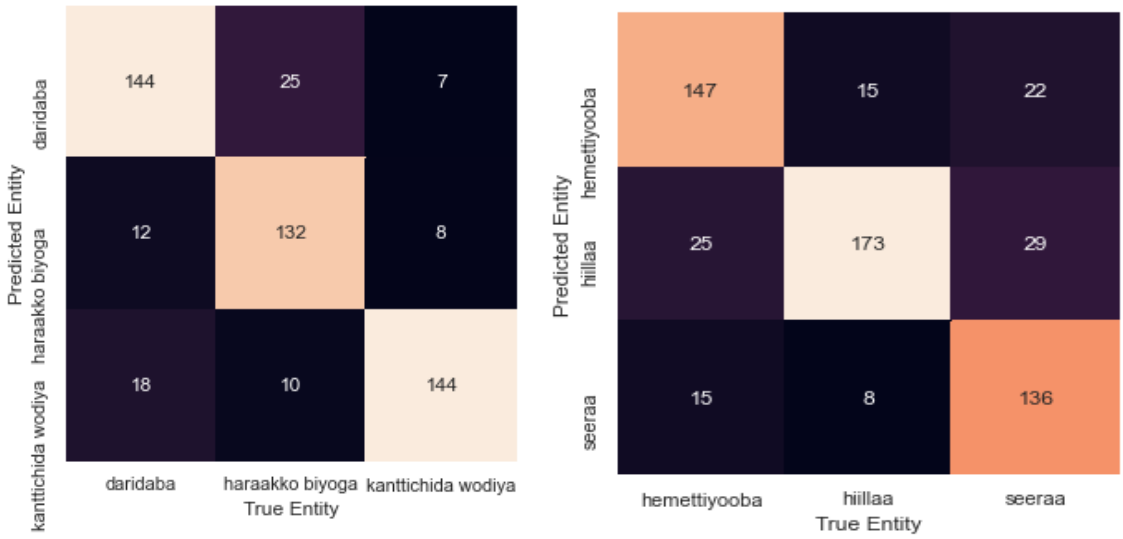



*Figure 6. 5 Result of SVC Confusion matrix using Left (Aadhdha dataset) and right (Ogiya dataset)*

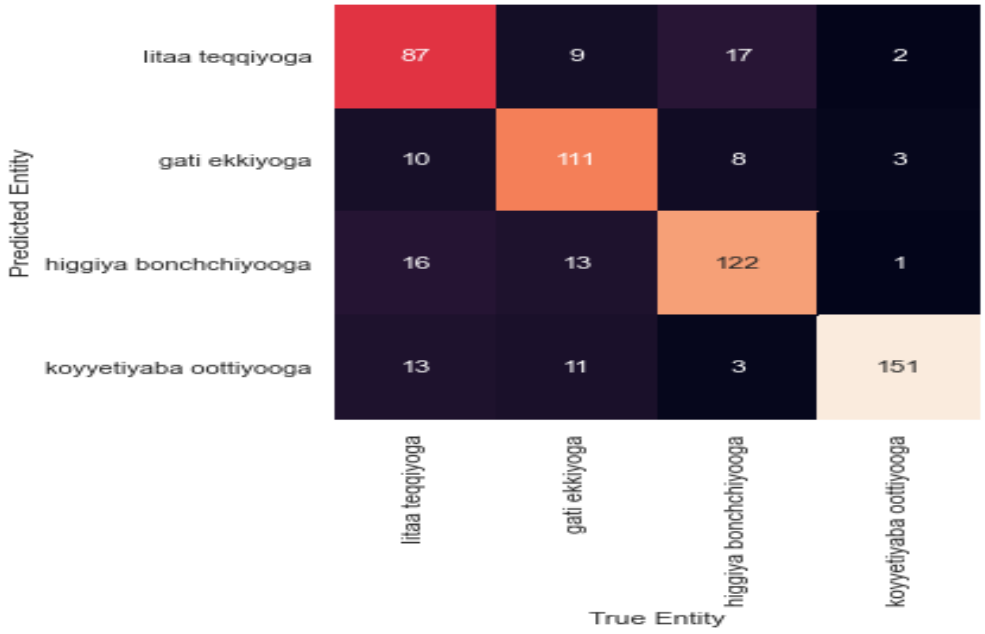



*Figure 6. 6 Confusion matrix result of SVM using Naaga Dataset*

As shown in the figures above, we evaluated five datasets using SVC. As can be seen from Figure 6.4, from the dataset (Doona), SVC classifies 427 sense examples correctly out of 500 sense examples, where only 73 sense examples are classified incorrectly. And also from the dataset (Ayfiya), 543 sense examples are classified out of 650 sense examples, where only 107 sense examples are classified incorrectly. As shown in Figure 6.5, from the dataset (Aadhdha), 420 sense examples classified correctly out of 500 sense examples, where only 80 senses are classified incorrectly. And also from the dataset (Ogiya), 456 sense examples classified correctly out of 570 sense examples, where only 114 sense examples classified incorrectly. I addition to these, as shown in Figure 6.6, from the dataset (Naaga), 471 sense examples classified correctly out of 577 sense examples, where only 106 sense examples classified incorrectly.

### *6.5.2 Confusion Matrix Result of Bagging Classifier using Five Datasets*

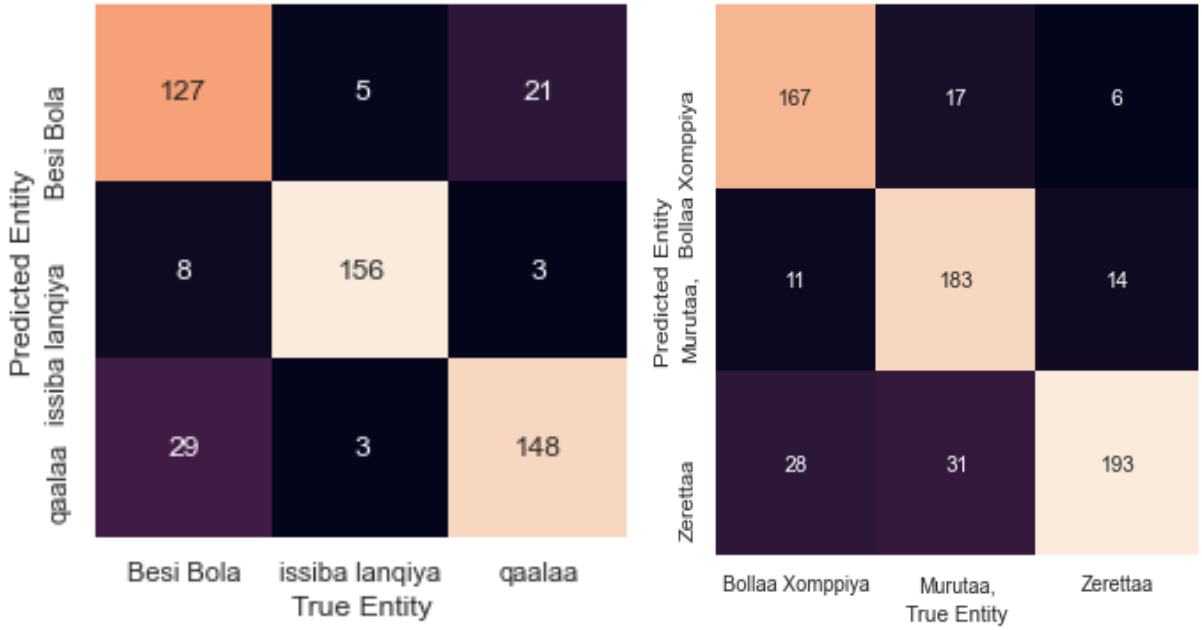



*Figure 6. 7 Result of Bagging Classifier Confusion matrix using Left (Doona dataset) and right (Ayfiya dataset)*

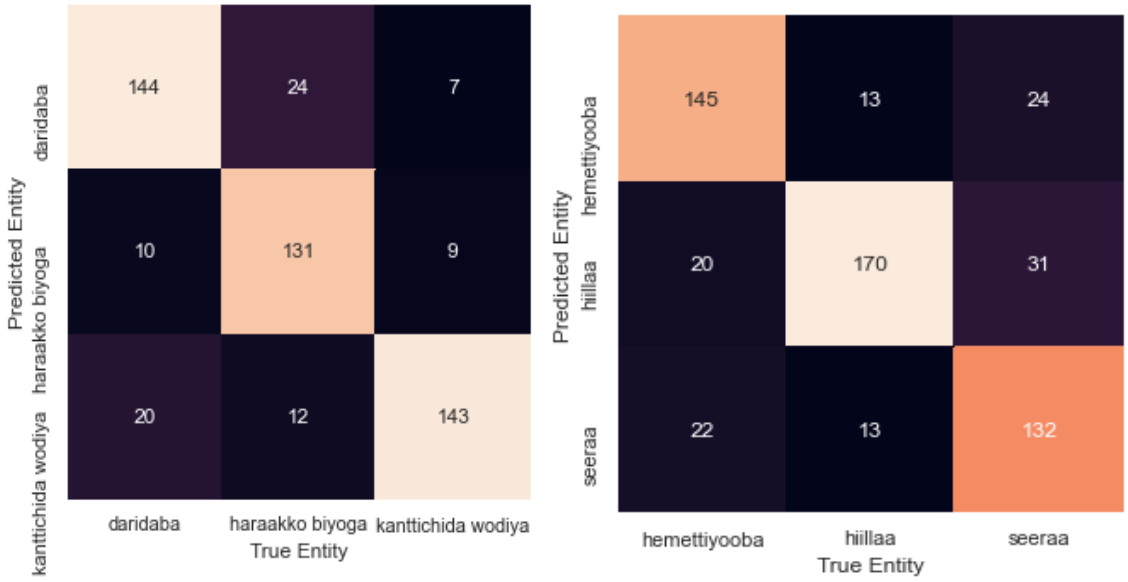



*Figure 6. 8 Result of Bagging Classifier Confusion matrix using Left (Naaga dataset) and right (Aadhdha dataset)*

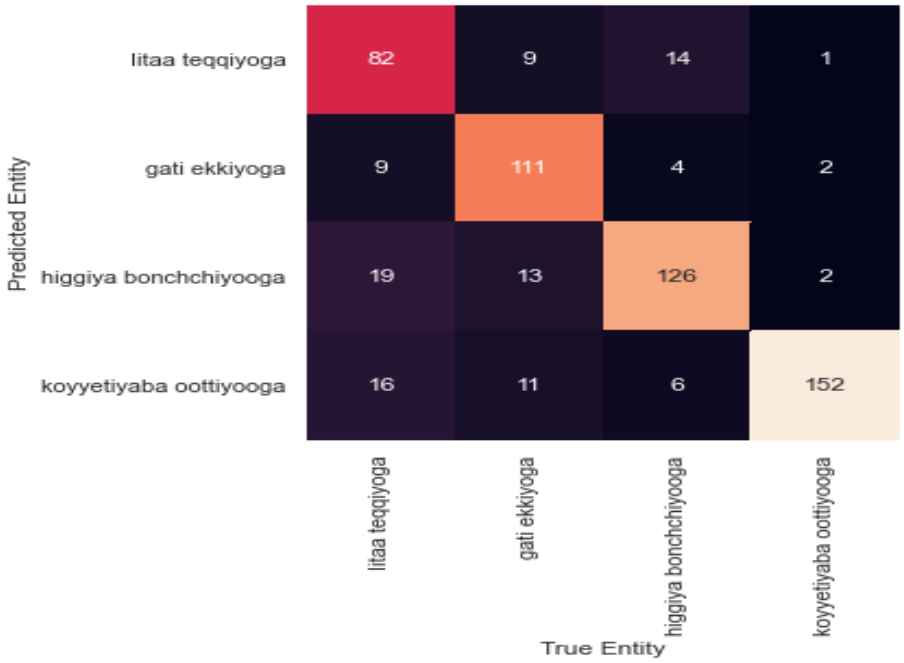



*Figure 6. 9 Confusion matrix result of Bagging Classifier using naaga Dataset*

As shown in the figures above, we evaluated five datasets using Bagging Classifier. As can be seen from Figure 6.7, from the dataset (Doona), Bagging classifies 431 sense examples correctly out of 500 sense examples, where only 69 sense examples are classified incorrectly. And also from the dataset (Ayfiya), 543 sense examples are classified out of 650 sense examples, where only 104 sense examples classified incorrectly. As shown in Figure 6.8, from the dataset (Aadhdha), 418 sense examples classified correctly out of 500 sense examples, where only 82 senses are classified incorrectly. And also from the dataset (Ogiya), 447 sense examples classified correctly out of 570 sense examples, where only 123 sense examples classified incorrectly. I addition to these, as shown in Figure 6.9, from the dataset (Naaga), 441 sense examples classified correctly out of 577 sense examples, where only 136 sense examples classified incorrectly.

*Table 6. 5 Summary of Evaluation performance Metrics of SVC and Bagging*

| ***SVC*** | ***Accuracy*** | ***Precision*** | ***Recall*** | ***F1-score*** |
|---|---|---|---|---|
| Doona | 85.4 | 85 | 85 | 85 |
| Ayfiya | 83.5 | 84 | 84 | 84 |
| Aadhdha | 84 | 84 | 84 | 84 |
| Naaga | 81.6 | 81 | 81 | 82 |
| Ogiya | 80 | 81 | 80 | 80 |
| Average | **82.9** | 82.6 | 82.2 | 82.2 |
| **Bagging** | | | | |
| Doona | 86.2 | 86 | 86 | 86 |
| Ayfiya | 83.5 | 83 | 83 | 83 |
| Aadhdha | 83.6 | 84 | 84 | 84 |
| **Naaga** | 81.6 | 82 | 81 | 82 |
| **Ogiya** | 78.4 | 80 | 79 | 79 |
| **Average** | **82.66** | 82.2 | 82 | 82 |

As shown in Table 6.5, the average accuracies achieved using SVC and Bagging Classifier are 82.9 % and 82.66 % respectively.

### *6.6.3 Model Evaluation Result Using User Evaluation*

In this section, we evaluated the model performance by inputting the new and unseen data gathered from Holy Bible, News, Academic data, and other spoken sentences. As mentioned in Appendix 8, the WSD prototype was developed to test our model by 45 sentences, and out of 45 sentences, the model predicted 38 senses and predicted 7 senses incorrectly. We calculated the accuracy of user evaluation by Equation (12).

$$\text{Accuracy} = \frac{total\ number\ of\ predicted\ senses(meaning) in\ the\ context}{The\ total\ number\ of\ expected\ meaning\ of\ user} \quad \textbf{(12)}$$

Based on these, our model achieved an accuracy of 84% in user data evaluation as mentioned in Appendix 9. This indicates that our model predicts 16% incorrectly. Therefore, the WSD model wolaita language is better to identify five polysemy words of Wolaita language.

## 6.6 Comparison with the existing WSD model

As the best knowledge of the researcher, there are no prior researches conducted for WSD of Wolaita language. In this section, we compared our study with recently conducted WSD of three languages such as Geez, Afan Oromo, and Tigrigna languages.

Firstly we compared our work with the WSD of Geez language (Amlakie Aschale et al., 2021), the objective of this research is to identify two senses for six polysemy words using ADTree and Bagging. The researcher selected the optimal window size for 2119 sentences based on the average accuracy of the model. But in our work, we selected optimal window size based on the average accuracy achieved and the low computational time the model costs. In addition to this, we identified three and four senses for polysemy words of the Wolaita language.

Secondly, we compared our work with the WSD for Tigrinya language ( Teklay, 2018). The researcher conducted WSD of Tigrinya language using ADTree and Bagging. He selected window one (1-1) for 1250 sentences and five ambiguous words to identify two up to four senses of the polysemy words. In his work, he only focused on the average accuracy achieved and small window size. But, smaller window sizes can also make the sense to be syntactic rather than semantic meaning. In our study, we used 2797 sentences and we selected optimal window size based on the average accuracy and the lower computational time cost. In addition to this, we identified three and four senses for polysemy words of the Wolaita language.

Thirdly we compared our work with the WSD of Afan Oromo (Tesema, 2017). In this works, the researcher used a clustering algorithm (vector Space Model) only for one ambiguous word 'Bahe'. As the above researchers, the researcher also selected window size (2-2) for a small amount of data based on the average accuracy achieved. In our study, we used 2797 sentences for five ambiguous words which were annotated by language experts and we selected optimal window size based on the average accuracy and the lower computational time cost.

The average accuracy achieved by our model is 83.22 and 82.9 using 10-fold CV and confusion matrix respectively. For time being it is enough and some of other WSD works have higher accuracy than our WSD. The reason behind this is, the gemination of vowels and consonants and elongation of suffixes in Wolaita texts make the morphology complex. And also, there is no stemming and lemmatization were applied in data preprocessing step since there are no freely available stemming and lemmatization on internet. Therefore, these can reduce the performance of the model. For time constraint, we omitted these things and these can increase the performance of the model if they will be applied in data preprocessing step in the future time.

## 6.7 Discussion

As we discussed in section 1.5.1, the general objective of this study is the WSD Model for the Wolaita language. To achieve the objectives, we have to answer the research questions discussed in section 1.4.

**RQ1**: Which machine learning algorithms best perform for WSD of Wolaita language? To answer this question, we selected the algorithms that work best in multi-class classification. These algorithms are SVC, RF, Bagging, and AdaBoost and we collected 2797 sense examples for five polysemy words that have three and four senses (meaning) according to the context. From the selected algorithms SVC and Bagging classifiers achieved better results based on performance evaluation metrics and user evaluation as discussed in section 6.6.3.

**RQ2**: How can the overall performance of WSD for Wolaita language be enhanced? To answer this question, we selected best feature extraction techniques called TF-IDF and L2 regularization technique and 10-fold CV used to handle overfitting problem. In addition to this, we selected optimal window size based on the highest average accuracy and the lower computational time cost. As discussed in section 6.4, window size 11(5-5) was selected as optimal window size for the WSD model of Wolaita language in terms of average accuracy and the computational time it costs.

**RQ3:** To what extent does the model disambiguate ambiguous words from Wolaita texts? To answer this question, we used the optimal window size that was discussed above. We evaluated the two models which achieved the higher average accuracy in optimal window size (WS11) in the above section 6.5. Then we evaluated these models using performance evaluation metrics and user evaluation. Based on these SVM and Bagging classifiers achieved an average accuracy of 82.9% and 82.66% respectively.

# Conclusion, Recommendation, and Future work

## Conclusion

In this research, we conducted WSD of Wolaita language using the Machine Learning Approach. In computational linguistics, polysemy words faced the difficulties of ambiguity for NLP applications such as MT, IR, speech processing, and other NLP applications. Therefore, identifying the senses (meaning) of polysemy words in the context increases the performance of the NLP applications. In this study, we researched the WSD of Wolaita language to enhance the performance of the NLP applications that were conducted by prior researchers and the researchers who will conduct NLP related researches.

In NLP, the knowledge-acquisition bottleneck (too many words, too many senses, and too many languages) is the problem in WSD. In our work, we selected five ambiguous words due to the time constraint and we collected 2797 sense examples to identify three and four senses in the context. To do that, we selected four classifiers such as SVC, Bagging, RF, and AdaBoost. We trained these models based on six-window sizes. Depending on the average accuracy achieved and the low computational cost, we selected WS11 (5-5). Based on WS11 (5-5), we evaluated our models. Finally, SVC and Bagging Classifier achieved higher accuracy. Therefore, we selected these classifiers as the models of WSD of the Wolaita language. Therefore choosing window size in terms of accuracy and the low computational time cost can play a vital role when a large amount of data supplied by users for the number of data is increasing on the internet.

In addition to this, we experimented on the base-estimators of the Bagging Classifier to improve the performance of the Bagging model. Based on the criteria discussed in section 6.3, Linear SVC with Bagging Classifier achieved higher accuracy. Therefore, instead of using a DT base estimator, using best weak learner can increase the performance of the Bagging Classifier in terms of accuracy.

As the best knowledge of the researcher, WSD for Wolaita language is the first attempt and it is used as data preprocessing for the NLP applications and It can increase the performance of the these applications. As the first researcher, we prepared datasets according to Sensival evaluation criteria and these can play the great role for the next researchers to use the datasets and to

increase the size of data and the number of ambiguous words. Due to the time constraint, we selected five ambiguous words.

**Recommendations**

From the study we conducted, we understood that integrating WSD with NLP application play a vital role to increase the performance of the application. In the following section, we recommend:

- ✓ **Researchers of NLP** to integrate WSD with the developed applications such as MT, Speech Recognition, Text-to-Speech synthesizer.
- ✓ **Researchers of NLP** to integrate the WSD with their newer NLP projects and researches to increase the performance of their applications.

**Future works**

In this study, due to the lack of sense annotated corpus mainly WordNet for Wolaita language, we prepared corpus manually and conducted corpus-based WSD. Therefore, we mentioned the following things that will be done in the future.

- ✓ Knowledge-Based WSD for Wolaita language using Deep learning approaches for more than five ambiguous words. To conduct this, it is important to develop WordNet using stakeholders such as Language experts, NLP experts, and other personnel who are important in this work.
- ✓ WSD model with spell, grammar checker, and POS tagger. Due to the unavailability of these works on the internet, developing all of these things for WSD is time and resource-consuming. Therefore, in the future, it is important to integrate all of these parts.
- ✓ Implementation of WSD model with Wolaita Stemmer and WordNet Lemmatization.

==========

**SPECIAL ACKNOWLEDGMENT**

This research work was funded by Adama Science and Technology University under grant number: ASTU/SM-R/240/21

Adama, Ethiopia

# REFERENCES


BEKELE, Y. T. (2016). Towards the Sense Disambiguation of Afan Oromo Words Using Hybrid Approach(Unsupervised Machine Learning and Rule Based). *Ethiop. J. Educ. & Sc., 12*.

Teklay, M. (2018). Word Sense Disambiguation of Tigirinya Using Semi-supervised learning. *AAU ETD*.

Adams, B. A. (n.d.). *Wolaytta language*. (Wikipedia, the free encyclopedia) Retrieved from https://en.wikipedia.org/wiki/Wolaytta_language

Alessio et al. (2014). Pragmatic Ambiguity Detection in Natural Language Requirements. *IEEE*.

Amlakie Aschale et al. (2021). Corpus-Based Word Sense Disambiguation for Ge'ez Language. *Ethiopian Journal of Science and Sustainable Development, 8*.

Andres Montoyo et al. (2005). Combining Knowledge- and Corpus-based Word-Sense-Disambiguation Methods. *Journal of Artificial Intelligence Research*, 299-330.

Anjali , M K1. (2014). Ambiguities in Natural Language Processing. *International Journal of Innovative Research in Computer and communication engineering, 2*(5).

Anjali , M., & Babu . (2014). Ambiguities in Natural Language Processing. *International Journal of Innovative Research in Computer and Communication Engineering*.

ASSEMU, S. (2011). UNSUPERVISED MACHINE LEARNING APPROACH FOR WORD SENSE DISAMBIGUATION TO AMHARIC WORDS. *AU ETD*.

Atnafu Lambebo et al. (2021). Multilingual Neural Machine Translation for Low Resourced Languages: Ometo-English. *Third International Conference on ICT for Development for Africa*.

Aung, y. T. (2011). A Word Sense Disambiguation System Using Naïve Bayesian Algorithm Myanmar Language. *International Journal of Scientific & Engineering Research, 2*(9).

Ba¸skaya, O., & Jurgens, D. (2016). Semi-supervised Learning with Induced Word Senses for State of the Art Word Sense Disambiguation. *Journal of Artificial Intelligence Research*.

Bekele, Y. T. (2016). Hybrid Word Sense Disambiguation Approach for Afaan Oromo. AA: AAU.

Berke Özen et al. (2017). All-word Word Sense Disambiguation for Turkish. *IEEE*.

Bianca Scarlini et al. (2020). SENSEMBERT: Context-Enhanced Sense Embeddings for MultilingualWord Sense Disambiguation. *Association for the Advancement of Artificial*.

Bin Li et al. (2010). An Investigation of Chinese Selectional Preference Based on HowNe. *IEEE*.

*Bootstrap aggregating.* (2021). Retrieved 6 17, 2021, from From Wikipedia, the free encyclopedia: https://en.wikipedia.org/wiki/Bootstrap_aggregating

Brown Keith et al. (2005). Encyclopedia of Language and Linguistics, Second Edition. *Elsevier*, 665-670. Retrieved 02 26, 2021, from https://wordnet.princeton.edu/

Bruce G. , M. (2021). What is an optimal value of k in k-fold cross-validation in discrete Bayesian network analysis? *Computational Statistics (*.

Calvo, H. (2018). Towards Universal Word Sense Disambiguation using Deep neural network. *IEEE*.

Choueka, Y. (2014). Natural Language Processing Challenges, Achievements and Problems . *Springer*.

Dagimawi, L. (n.d.). *Wolaita-English Dictionary.* Retrieved from https://play.google.com/store/apps/details?id=com.laxsil.dagmawi.woliticdictionery&hl=en_US&gl=US

Dan TUFIŞ, R. I. (2003). WORD SENSE CLUSTERING BASED ON TRANSLATION EQUIVALENCEIN PARALLEL TEXTS; A CASE STUDY IN ROMANIAN. *Romanian Academy*.

Daniel Jurafsky & James H. Martin. (2006). *Speech and Language Processing: An introduction to natural language processing,computational linguistics, and speech recognition.*

Dehaibi et al. (2020). Investigating Inter-rator Reliablity of quality text annotation in machine learning datasets. *International Design conferance*.

Dian Paskalis, F. B. (2011). Information Retrieval Using Query Expansion. *International Conference on Electrical Engineering and Informatics,IEEE*.

Dongsuk O, e. a. (2018). Word Sense Disambiguation Based on Word Similarity Calculation Using Word Vector Representation from a Knowledge-based Graph. *27th International Conference on Computational Linguistics*, 2704–2714.

Edi Faisal et al. ( 2018). Word Sense Disambiguation in Bahasa Indonesia Using SVM. *IEEE*.

Eneko Agirre and Philip Edmonds. (2007). Word Sense Disambiguation Algorithms and Applications. *Springer, 33*.

Eneko Agirre,German Rigau. (2014). Word Sense Disambiguation using Conceptual Density. *Computation and Language (cs.CL)*.

Eneko Agirree and Philip Edmonds. (2007). Word Sense Disambiguation Algorithms and Applications. *Springer*.

Faiza , M., & Mohammed , A. (2021). Features Extraction Effect on the Accuracy of Sentiment Classification Using Ensemble Models. *International Journal of Science and Research (IJSR)*.

Fanta, H. (2010). SPEAKER DEPENDENT SPEECH RECOGNITION FOR WOLAITA LANGUAGE. *AAU REPOSITORY* . Retrieved from http://etd.aau.edu.et/handle/123456789/14582

Fauzi, M. A. (2019). Word2Vec model for sentiment analysis of product reviews in Indonesian language. *International Journal of Electrical and Computer Engineering (IJECE)*.

From Wikipedia, t. f. (n.d.). *Multinomial logistic regression.* Retrieved from https://en.wikipedia.org/wiki/Multinomial_logistic_regression

From Wikipedia, t. f. (n.d.). *Word-sense disambiguation.* Retrieved May 2021, from https://en.wikipedia.org/wiki/Word-sense_disambiguation

G., B. (2020). POS Tagger for Wolaita Using Transformational Based Algorithms. *Research Gate*.

Gebreselassie, T. A. (2018). A Finite-State Morphological Analyzer for Wolaita. *LNICST*.

Gebresilase, T. A. (November 29, 2013). TEXT TO SPEECH SYNTHESIZER FOR WOLAYTTA LANGUAGE. *LAP Lambert Academic Publishing* . Retrieved from https://scholar.google.com/scholar?hl=en&as_sdt=0%2C5&q=TEXT-TO-SPEECH+SYNTHESIZER+FOR+WOLAYTTA+LANGUAGE&btnG=

GEOFFREY I, e. a. (2001). Machine Learning for User Modeling. *Kluwer Academic Publishers*.

Gerard et al. (2000). A Comparison between Supervised Learning Algorithms for Word Sense Disambiguation . *4th conference on Computational natural language learning*.

Getahun Wassie and Solomon Tefera. (2014). A Word Sense Disambiguation Model for Amharic Words using Semi-Supervised Learning Paradigm. *Science, Technology and Arts Research Journal*.

Getahun Wassie et al. (2014). A Word Sense Disambiguation Model for Amharic Words using Semi-Supervised Learning Paradigm. *Science, Technology and Arts Research Journal*.

Getaneh, M. (2020). Automatic wordNet construction using word embedding. *AAU ETD*.

Girma, Y. (2018). Development of Longest-Match Based Stemmer for Texts of Wolaita Language. *International Journal on Data Science and Technology*.

Gonfa, S. O. (2018). WORD SENSE DISAMBIGUATION FOR AFAAN OROMO USING KWNOWLEDGE BASE. *ST.MARRY UNIVERSITY*.

Great Learning Team. (2020). *The Ultimate Guide to AdaBoost Algorithm | What is AdaBoost Algorithm?* (GreatLearning) Retrieved from https://www.mygreatlearning.com/blog/adaboost-algorithm/

Hristea, F. T. (2011). Statistical Natural Language Processing. *International Encyclopedia of Statistical Science*(ISBN : 978-3-642-04897-5).

Hundesa, Tesfa Kebede. (2013). WORD SENSE DISAMBIGUATION FOR AFAAN OROMO LANGUAGE. *AU repository*.

Hundesa, T. K. (2013). WORD SENSE DISAMBIGUATION FOR AFAAN OROMO LANGUAGE . *AAU ETD*.

Johansson et al. (2015). Combining Relational and Distributional Knowledge for Word Sense Disambiguation. *Proceedings of the 20th Nordic Conference of Computational Linguistics (NODALIDA 2015)*.

Jumi Sarmah et al. (,2016). Decision Tree based Supervised Word Sense Disambiguation for Assamese. *International Journal of Computer Applications (0975 – 8887), 141*.

Jurafsk, D. (2019). *Speech and Language Processing.* Stanford.

Kalita, P., & Barman. (2015). Implementation of Walker algorithm in Word Sense Disambiguation for Assamese language. *IEEE*.

*Lesk_algorithm*. (2021). (Wikipedia, the free encyclopedia) Retrieved 02 25, 2021, from Wikipedia: https://en.wikipedia.org/wiki/Lesk_algorithm

Lessa, L. (2003). Development of Stemming Algoritm for Wolaita Language. *AAU ETD*.

Lu, W. (2019). Graph-Based Chinese Word Sense Disambiguation with Multi-Knowledge Integration. *CMC, 61*.

*Machine-readable_dictionary.* (2021). Retrieved 02 25, 2021, from Machine-readable dictionary,from wikpedia: https://en.wikipedia.org/wiki/Machine-readable_dictionary

Mekonnen, S. (2010). WORD SENSE DISAMBIGUATION FOR AMHARIC TEXT: A MACHINE LEARNING APPROACH. *AU ETD*. Retrieved from http://localhost:80/xmlui/handle/123456789/14751

Mohammad Shibli Kaysar, e. a. (2019). Word Sense Disambiguation of Bengali words using FP-Growth. *IEEE*.

Morphology of Wolaita. (2021). *Unpublished work*.

Mrs. A. S. More and Dr. Dipti P. Rana. (2017). Review of Random Forest Classification Techniques to Resolve Data Imbalance. *IEEE*.

Mulugeta, M. (2020). Word Sense Disambiguation for Amharic Sentences using wordnet hierarchy. *BU ETD*.

*Naive Bayes classifier*. (2021). Retrieved 3 22, 2021, from From Wikipedia, the free encyclopedia: https://en.wikipedia.org/wiki/Naive_Bayes_classifier

*Natural language processing.* (2021). Retrieved from From Wikipedia, the free encyclopedia: https://en.wikipedia.org/wiki/Natural_language_processing

NG'ANG'A, W. (2005). Word Sense Disambiguation of Swahili:Extending Swahili Language Technology with Machine Learning. *Faculty of Arts*.

Nigatu, E. (2021). Fake News Detection For Amharic Language Using Deep Learning. *ASTU-ETD*.

Osman, B. (2016). Semi-supervised Learning with Induced Word Senses for State of the Art Word Sense Disambiguation. *Journal of Artificial Intelligence Research*.

Pedersen. (2007). Unsupervised Corpus-Based Methods for WSD. *Springer, 33.*

Pekins, J. (2010). *Python Text Processing with NLTK 2.0 Cookbooks.* UK: Packt Publishing.

Philip, E., & Eneko , A. (2008). *Word_sense_disambiguation.* Retrieved from http://www.scholarpedia.org/article/Word_sense_disambiguation

Pinar Y, e. a. (2005). The Effect of Windowing in WSD. *ISCiS*.

Popov, A. (2018). Neural Network Models for Word Sense Disambiguation:Overview. *CYBERNETICS AND INFORMATION TECHNOLOGIES, 18*, .

Pranjal P. et al. (March 2014). Approaches for Word Sense Disambiguation-survey. *International Journal of Recent Technology and Engineering (IJRTE), 3*(1).

Preeti Rana and Parteek Kumar. (2015). Word Sense Disamiguation for Punjabi Language using Overlap Based Approach. *Springer, 320*, 607.

Reda, M. M. (2018). Unsupervised Machine Learning Approach for Tigrigna Word Disamiguation. *Computer Engineering and Intelligent Systems, 9*, 6.

Reda, Meresa Mebrahtu. (2018). Unsupervised Machine Learning Approach for Tigrigna WSD. *Computer Engineering and Intelligent Systems ISSN 2222-1719 (Paper) ISSN 2222-2863 (Online), 9*, 6.

Resham N. Waykole1, Anuradha D. Thakare2. (2018). A REVIEW OF FEATURE EXTRACTION METHODS FOR TEXT CLASSIFICATION. *International Journal of Advance Engineering and Research Development, 5*(4).

Robert Dzisevič,Dmitrij Šešok. (2019). Text Classification using Different Feature Extraction Approaches. *IEEE*.

Rui Suzuki, e. a. (2018). All-words Word Sense Disambiguation Using Concept Embeddings. *National Institute for Japanese Language and Linguistics*.

*SemEval*. (2021). (Wikipedia, the free encyclopedia) Retrieved from Wikipedia: https://en.wikipedia.org/wiki/SemEval

Shirko, B. F. (2020). Part of Speech Tagging for Wolaita Language using Transformation Based Learning (TBL) Approach. *IJESC, 10*(9).

Shubham Kumar et al. (2019). Application of Natural Language Procesing and IoTCloud in Smart Homes. *IEEE*.

Siddique et al. (2021). Machine Learning in healthcare. *Encyclopedia*.

Siddiqui, S. S. (2012). Evaluating Effect of Context Window Size, Stemming and Stop Word Removal on Hindi Word Sense Disambiguation. *IEEE*.

Tamara Martin. (2015). The clustering-based Approach for Unsupervised Word Sense Disambiguation. *Sociedad Española para el Procesamiento del Lenguaje Natural*.

Tesema, W. (2017). Word Sense Disambiguation and Semantics for Afan Oromo Words using Vector Space Model. *International Journal of Research Studies in Science, Engineering and Technology, 4*(6), 6.

Teshome. (2009). Word Sense Disambiguation for Amharic Text Retrieval. *AAU ETD*.

Ting, K. M. (2011). Confusion Matrix. *Webb G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston*.

*Toophphiya Hayo Bilaa Woshettaa.* (2007). Wolaita Sodo: Unpublished.

Toophphiya Hayyo Billaa Woshettaa. (2008). *Unpublished*.

Tutorialspoint. (n.d.). *natural_language_processing.* Retrieved Feb 20, 2021, from https://www.tutorialspoint.com/natural_language_processing/natural_language_processing_introduction.htm

Udaya Raj Dhungana et al. (2015). Word Sense Disambiguation using WSD Specific WordNet of Polysemy Words. *IEEE 9th International Conference on Semantic Computing*.

Wahab , Khan1. (2016). A survey on the state-of-the-art machine learning models in the context of NLP. *Kuwait J. Sci*.

WAKASA, M. (2014). A Sketch Grammar ofWolaytta. *Nilo-Ethiopian Studies*.

*Warren Weaver*. (2021). Retrieved from Wikipedia, the free encyclopedia: https://en.wikipedia.org/wiki/Warren_Weaver

*Warren_Weaver*. (2021). Retrieved Feb 20, 2021, from WIkpedia, Free encyclopedia: https://en.wikipedia.org/wiki/Warren_Weaver

WELEMICHAEL, T. M. (2018). WSD for Tigrinya Language using Semi-supervised learning. *AAU ETD*.

*Wolaitta language.* (2021). Retrieved from From Wikipedia, the free encyclopedia: https://en.wikipedia.org/wiki/Wolaitta_language

*Word Sense Disambiguation.* (2021). Retrieved 02 25, 2021, from Wikipedia, the free encyclopedia: https://en.wikipedia.org/wiki/Word-sense_disambiguation#Dictionary-_and_knowledge-based_methods

*Word Sense Disambiguation.* (2021, Feb 4). Retrieved Feb 20, 2021, from From Wikipedia, the free encyclopedia: https://en.wikipedia.org/wiki/Word-sense_disambiguation

*Word-sense disambiguation.* (2021). Retrieved from Wikipedia, the free encyclopedia: https://en.wikipedia.org/wiki/Word-sense_disambiguation#History

Yang, H. (2011). Analysing anaphoric ambiguity in natural language requirements. *Springer-Verlag*.

*Yehoshua Bar-Hillel.* (2021). Retrieved Feb 20, 2021, from From Wikipedia, the free encyclopedia: https://en.wikipedia.org/wiki/Yehoshua_Bar-Hillel

Yesuf, S. H. (2015). AMHARIC WORD SENSE DISAMBIGUATION USING WORDNET. *AAU ETD*.

Ying, X. (2018). An Overview of Overfitting and its solution. *Journal of Physics: Conf. Series*, 4.

# APPENDIXES

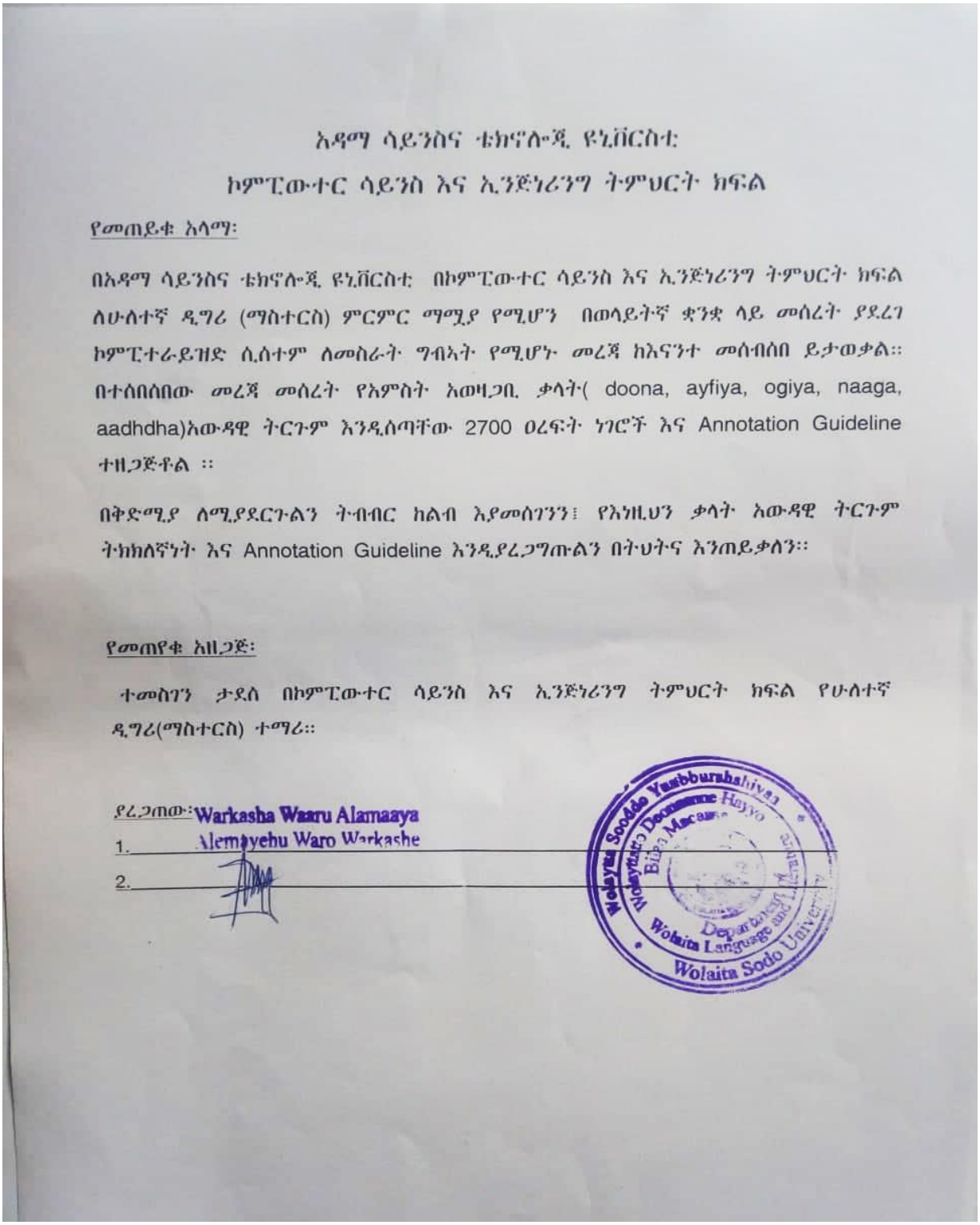
## Appendix 1: Data Annotation Guideline for WSD Corpus for Wolaita Language

Data annotation guideline is aimed to enhance the quality of the dataset which is trained by model. In our study, data was collected from different sources. Therefore, annotating these data requires language experts. Despite, language experts annotate reliable data, they don't know about the way how the dataset is prepared. Therefore, we prepared a data annotation guideline depending on Sensival evaluation. In our work, we have selected five ambiguous words such as 'Naaga','aadhdha','ogiya' ,'doonaa', and 'ayfiya'.

The following are annotations guidelines that are based on Sensival evaluations (SemEval, 2021).

1. Use Microsoft excel to annotate data
2. Don't annotate the sentences that don't contain the five ambiguous words
3. For each ambiguous word, annotate their sense examples(important features)
4. assign the same label to the same class
5. annotate the same number of sense examples
6. Check grammar and spelling error

**Appendix 2: A Confirmation of Data Annotation by Wolaita Language and Literature Department**



አዳማ ሳይንስና ቴክኖሎጂ ዩኒቨርስቲ

ኮምፒውተር ሳይንስ እና ኢንጅነሪንግ ትምህርት ክፍል

የመጠይቁ አላማ፡

በአዳማ ሳይንስና ቴክኖሎጂ ዩኒቨርስቲ በኮምፒውተር ሳይንስ እና ኢንጅነሪንግ ትምህርት ክፍል ለሁለተኛ ዲግሪ (ማስተርስ) ምርምር ማማያ የሚሆን በወላይትኛ ቋንቋ ላይ መሰረት ያደረገ ኮምፒተራይዝድ ሲስተም ለመስራት ግብኣት የሚሆኑ መረጃ ክአናንተ መሰብሰብ ይታወቃል። በተሰበሰበው መረጃ መሰረት የአምስት አወዛጋቢ ቃላት( doona, ayfiya, ogiya, naaga, aadhdha)አውዳዊ ትርጉም እንዲሰጣቸው 2700 ዐረፍት ነገሮች እና Annotation Guideline ተዘጋጅቶል ::

በቅድሚያ ለሚያደርጉልን ትብብር ከልብ እያመሰገንን፤ የእነዚህን ቃላት አውዳዊ ትርጉም ትክክለኛነት እና Annotation Guideline እንዲያረጋግጡልን በትህትና እንጠይቃለን።

የመጠየቁ አዘጋጅ፡

ተመስገን ታደሰ በኮምፒውተር ሳይንስ እና ኢንጅነሪንግ ትምህርት ክፍል የሁለተኛ ዲግሪ(ማስተርስ) ተማሪ።

ያረጋጠው፡ Warkasha Waaru Alamaaya

1. Alemayehu Waro Warkashe

2.

Wolaita Sodo University Bibra Macaa Hayyo Department of Wolaita Language and Literature Wolaita Sodo University

## Appendix 3: IRR measurement of four annotators using Fleiss' Kappa

This appendix indicates that the overall agreement between four annotators and agreement on individual categories (the number of sentences for 15 senses of ambiguous words).

### Fleiss Multirater Kappa

**Overall Agreement[a]**

| | Kappa | Asymptotic Standard Error | Asymptotic z | Asymptotic Sig. | Asymptotic 95% Confidence Interval Lower Bound | Asymptotic 95% Confidence Interval Upper Bound |
|---|---|---|---|---|---|---|
| Overall Agreement | .819 | .030 | 27.153 | .000 | .817 | .821 |

a. Sample data contains 15 effective subjects and 4 raters.

**Agreement on Individual Categories[a]**

| Rating Category | Conditional Probability | Kappa | Asymptotic Standard Error | Asymptotic z | Asymptotic Sig. | Asymptotic 95% Confidence Interval Lower Bound | Asymptotic 95% Confidence Interval Upper Bound |
|---|---|---|---|---|---|---|---|
| 126.00 | .067 | 1.000 | .105 | 9.487 | .000 | .993 | 1.007 |
| 144.00 | .067 | 1.000 | .105 | 9.487 | .000 | .993 | 1.007 |
| 150.00 | .083 | .782 | .105 | 7.417 | .000 | .775 | .788 |
| 159.00 | .050 | .649 | .105 | 6.158 | .000 | .643 | .656 |
| 164.00 | .133 | 1.000 | .105 | 9.487 | .000 | .993 | 1.007 |
| 166.70 | .017 | -.017 | .105 | -.161 | .872 | -.024 | -.010 |
| 167.00 | .050 | .649 | .105 | 6.158 | .000 | .643 | .656 |
| 172.00 | .067 | 1.000 | .105 | 9.487 | .000 | .993 | 1.007 |
| 173.00 | .017 | -.017 | .105 | -.161 | .872 | -.024 | -.010 |
| 174.00 | .050 | .649 | .105 | 6.158 | .000 | .643 | .656 |
| 187.00 | .133 | 1.000 | .105 | 9.487 | .000 | .993 | 1.007 |
| 195.60 | .017 | -.017 | .105 | -.161 | .872 | -.024 | -.010 |
| 196.00 | .050 | .649 | .105 | 6.158 | .000 | .643 | .656 |
| 205.50 | .017 | -.017 | .105 | -.161 | .872 | -.024 | -.010 |
| 206.00 | .050 | .649 | .105 | 6.158 | .000 | .643 | .656 |
| 213.00 | .067 | 1.000 | .105 | 9.487 | .000 | .993 | 1.007 |
| 231.00 | .067 | 1.000 | .105 | 9.487 | .000 | .993 | 1.007 |

a. Sample data contains 15 effective subjects and 4 raters.

## Appendix 4: A Sample Dataset for WSD for Wolaita Language

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| beetemekidesi | shuchchay | katamatu | ogiyaa | doonan | laaletti | | | uttiis | issiba lanqiya |
| beeteleme | giyo | efraata | ogiyaa | doonan | | | | moogettasu | issiba lanqiya |
| | beni | mayzzatu | gi'iize | doonay | Toophphiyan | pitaliyan | | xaafettiis | qaalaa |
| | Goday | biitaykka | ta | doonaa | qaalaa | siyo | yaagidi | yoottiis | bola kifiliya |
| macaasiyaa | kammiyobaa | ekkada | ollaa | doonaa | ba | maayuwaa | | hiixxaasu | issiba lanqiya |
| carkkoy | issi | haruuruwaa | abbaa | doonan | nuna | un"ettiyoga | doommidi | aggiis | issiba lanqiya |
| coo | haasaya | woykko | bolla | doonan | nuuni | siiqettanawu | | bessenna | bola kifiliya |
| | | Taani | oge | doonan | de'iya | shooshshaa | demmada | woraas | issiba lanqiya |
| | Xoossay | daawite | a | doonan | geeshsha | ayyannaa | | wottiis | bola kifiliya |
| daro | doonaa | eranchchati | yootiogaadar | doonay | asaa | oonnatettaa | | malaatees | qaalaa |
| digiriya | xekkan | loohissiyagetus | tamaarissiyag | doonay | laamettiyo | hanottaa | xeelliyaagan | pilggidosona | qaalaa |
| | | | | doona | giisiyay | haasayin | immiyo | birshshettaa | qaalaa |

**Appendix 5. List of Stopwords of Wolaita Language**

| | | | | |
|---|---|---|---|---|
| aagaa | awu | aybaassi | gaasoy | hayyana |
| aagaadan | awude | aybaawu | garsaaana | he |
| aageta | ay | aybaayo | garssa | hegaa |
| aako | ayba | aybbanne | garssan | hegaappe |
| aana | aybaa | aybee | gidana | hegankka |
| aappunee | aybaakka | aymala | gidikokka | hegassi |
| aappuni | aybaassi | aymalaa | gidikonne | hegeeti |
| aara | aybaawu | aynnekka | giddon | hegeetu |
| aaro | aybaayo | ayssi | giishin | hegetane |
| akko | aybbanne | ba | gishshaawu | hekko |
| ane | aybee | baa | guutta | henna |
| ani | aymala | baga | guyaan | i |
| appe | aymalaa | banta | guyiyan | iigaa |
| appekka | aynnekka | bau | ha | iigaadan |
| asati | ayssi | bawu, | ha"ika | iigeta |
| assa | ba | bollan | haa | iira |
| awa | baa | chii | hagaa | iiro |
| awaara | baga | cora | hagaadan | iko |
| awan | banta | daro | hagaakko | inteegaa |
| awani | bau | darotoo | hagaani | inteegeta |
| aaro | bawu, | eeeta | hagara | inteero |
| akko | bollan | eeeta | hagee | intekka |
| ane | chii | eetaagaa | hageeta | intessi |
| ani | awu | eetaageta | hageeti | intte |
| appe | awude | eetaaro | ha'i | intteero |
| appekka | ay | eta | hanna | issi |
| asati | ayba | etaagadan | hara | issibaanne |
| assa | aybaa | etaara | haray | issippe |
| awa | aybaakka | eti | hayya | issoy |

kase
keena
kuma
loitti
loytti
mala
mantta
mata
matan
mati
mule
mulekka
muliya
na'ee
ne
nee
neegaa
neegaadan
neegeta
neeni
neero
neessi
neeyyo
nena
nenaara
nenan
niyo
nu
nuna
nuugaa
nuugaadan
nuugeta
nuukko
nuuni
nuura
nuuro
nuusi
nuuyoo
o
oogee
ookko
oonara
oonne
oonninne
oossi
ooyoo
qassi
qassikka
shin
simin
simmi
ta
taagan
taagaa
taagaadan
taageta
taana
taani
taassa
taassi
taayoo
taayyo
tabaa
taban
tana
tani
tawu
ubbaappe
ubbagallasi
ubban
ubbawode
ubbayoho
uray
wode
woqqu
woykko
ya
yaa
yaagidi
yaagin
yaako
yaanidaashin
yaanin
yaatin
yootin
yaanidaashin
yaanin

**Appendix 6. A Sample Code of implementation of model and Evaluation of the model**

```
# The implementation of Base_estimator or Weak Learner in Ensemble Methods
clf1 =clf = DecisionTreeClassifier(class_weight='balanced',max_depth=100)
clf2 = LinearSVC(penalty = 'l2',class_weight='balanced', C = 100, dual = False)
clf3 = SGDClassifier(loss="hinge", max_iter=200,penalty="l2")
clf4= LogisticRegression(n_jobs=1,multi_class='ovr',class_weight='balanced',solver='liblinear',C=1e5)
```

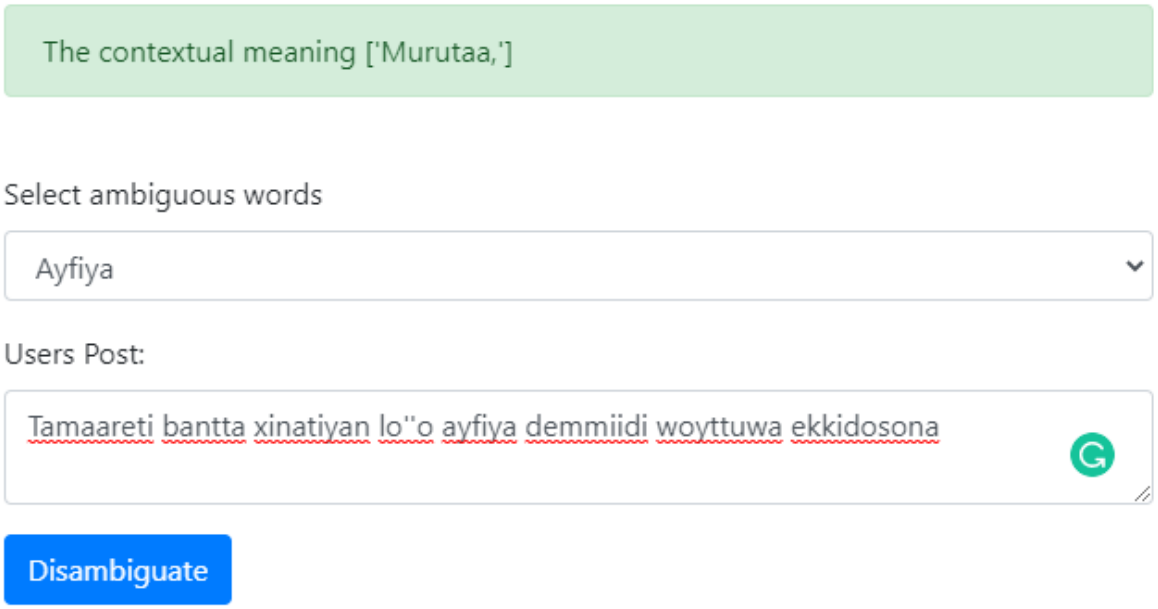
```
%%time
bag = Pipeline([('vect', CountVectorizer(ngram_range=(1, 3))),
                ('tfidf', TfidfTransformer()),
                 ('clf',  BaggingClassifier(n_estimators =100, base_estimator=clf3))])
bag.fit(WS9, y)
filename = 'doona_bag.sav'
pickle.dump(bag, open(filename, 'wb'))

CVscore=cross_val_score(bag, WS9, y, cv=10)
y_pred = cross_val_predict(bag, WS9, y, cv=10)
print('CV accuracy %s' % CVscore.mean())
print('accuracy %s' % accuracy_score(y_pred, y))
print(classification_report(y, y_pred))
```

## Appendix 7. A Sample Code for loading pickle data

```
import pickle
import numpy as np
def SVM_doona(data):
    nb = pickle.load(open('doona_svm.sav', 'rb'))
    data.lower()
    data=[data]
    pr=nb.predict(data)
    arr=nb.predict_proba(data)
    arr1=arr.tolist()
    confidence=np.amax(arr1)
    if confidence<0.8:
        return "I dont Understand"
    else:
        return pr
    return pr
def SVM_ayfiya(data):
    nb = pickle.load(open('ayfiya_svm.sav', 'rb'))
    data.lower()
    data=[data]
    pr=nb.predict(data)
    arr=nb.predict_proba(data)
    arr1=arr.tolist()
    confidence=np.amax(arr1)
    if confidence<0.8:
        return "I dont Understand"
    else:
        return pr
    return pr
```

**Appendix 8. User Interface Design of WSD Model of Wolaita language**



## Appendix 9. The Model Evaluation Result using Unseen Data

| | | | | |
|---|---|---|---|---|
| 35 | taani godaa hiddottan naagaas. | naaga | gati ekkiyoogaa | 1 |
| 36 | Piresidantte Puttiini a laggee Bayddeeni yaana gakkanaashin naagiis. | naaga | gati ekkiyoogaa | 1 |
| 37 | asay kaamiyan kumana gakkanaashin naagaydda takkaas. | naaga | gati ekkiyoogaa | 1 |
| 38 | Wolaita Dichay Geeshsha Gorggisiyara naagetiya kaassaykka de'ees | naaga | gati ekkiyoogaa | 0 |
| 39 | nu mayzza woga yelettaakko aattanawu nuuppe naagettiyaaba oottana koshshees. | naaga | koyyetiyaba oottiy | 1 |
| 40 | wolaytta dichay xoonnidi simmanaagee kassanchchatuppe naagettees. | naaga | koyyetiyaba oottiy | 1 |
| 41 | minnidi oottanaagee nuuppe naagettiyyoogaa cimati zoridosona. | naaga | koyyetiyaba oottiy | 1 |
| 42 | hiyyeessatettaa  xayssiyo ogiyan eridi minnidi oottana bessees. | ogiya | hiillaa | 1 |
| 43 | ne de'uwaa banqqayanawu cinccatettaa ogeta kaallana bessees. | ogiya | hiillaa | 1 |
| 44 | payyatettaa naagiyoo ogeta erana bessees. | ogiya | hiillaa | 1 |
| 45 | higge giddenna ogiyan zal'iyoogee qashissana danddayyees. | ogiya | seeraa | 1 |
| 46 | neeni ne xoossay biyo ogiya baana dandddayyabaakka. | ogiya | seeraa | 0 |
| 47 | Ne aaway biido ogiyan baadasa. | ogiya | seeraa | 1 |
| 48 | nuuni bazzo efiya ogiyaa oyqqidi soo yiida. | ogiya | hemettiyooba | 1 |
| 49 | hawaasaa efiya ogiyaa oiqqidi eti badeessi biidosona | ogiya | hemettiyooba | 1 |
| 50 | mehiyaa laaggiya zal''anchchati he ogiyaa aggidi loossuwaara biidosona | ogiya | hemettiyooba | 1 |
| 51 | | | | No. of 1= 42. |
| 52 | | | | No. of 0 = 8 |
| 53 | | | | Total=50 |
| 54 | | | Accuracy: | 42/50= 84% |
| 55 | | | | |