

A Comparative Analysis of Machine Learning Algorithms for Word Sense Disambiguation: In the Case of Wolaita Language



Destaye Ukumo Wada

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

June 2023

Adama, Ethiopia

A Comparative Analysis of Machine Learning Algorithms for Word Sense Disambiguation: In the Case of Wolaita Language

Destaye Ukumo Wada

Advisor

Sintayehu Hirpassa (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
Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

June 2023

Adama, Ethiopia

Declaration

I hereby declare that this Master Thesis entitled “**A Comparative Analysis of the Machine Learning Algorithms for WSD: In the Case of Wolaita Language**” 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.

Destave Ukumo Wada

Name of 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 “**A Comparative Analysis of the Machine Learning Algorithms for WSD: In the Case of Wolaita Language**” prepared under my/our guidance by Destaye Ukumo submitted in partial fulfillment of the requirements for the degree of Masters of Science in Computer Science and Engineering. Therefore, I/we recommend the submission of a revised version of the thesis to the department following the applicable procedures.

Dr. Sintayehu Hirpassa

Major Advisor

Signature

Date

Co-advisor

Signature

Date

Approval Page of M.Sc. Thesis

I/we, the advisors of the thesis entitled “**A Comparative Analysis of the Machine Learning Algorithms for WSD: In the Case of Wolaita Language**” and developed by Destaye Ukumo, hereby certify that the recommendation and suggestions made by the board of examiners are appropriately incorporated into the final version of the thesis.

Dr. Sintayehu Hirpassa

Major Advisor

Signature

Date

Co-advisor

Signature

Date

We, the undersigned, members of the Board of Examiners of the thesis by Destaye Ukumo have read and evaluated the thesis entitled “**A Comparative Analysis of the Machine Learning Algorithms for WSD: In the Case of Wolaita Language**” 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 and Engineering.

Chairperson

Signature

Date

Internal Examiner

Signature

Date

External Examiner

Signature

Date

Finally, approval and acceptance of the thesis is contingent upon the 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

I want to start by expressing my gratitude to my All-Powerful God for providing me with health, patience, and the ability to accomplish anything.

Next, I want to sincerely thank my advisor, Dr. Sintayehu Hirpassa, for all of his help with this study, including his advice, patience, understanding, promptness, great supervision, and fatherhood treatments. In addition to his intellectual counsel, he has taught me the meaning of humanity.

Besides my advisor, I would like to thank Mr. Temesgen Tadesse for supporting me in both moral and material aspects. And also supporting me in data collection and corpus preparation.

Also, I'd want to express sincere gratitude to the Artificial Intelligence Special Interest Group Members and CSE department MSc students who have become my inspirations by keeping me in harmony.

Also, I would like to express my gratitude to my friends in Adama Science and Technology for being with me at any time. And also want to thank my friends in Wolaita who inspired me to finish this work.

Last but not least, I want to express my profound gratitude to my dear parents and cousin Yohanis Boloshe (John) for their prayers and moral support. May God bless the with health and always protect them.

TABLE OF CONTENTS

ACKNOWLEDGMENTS	vi
LIST OF FIGURES	x
LIST OF TABLES	xi
ABBREVIATIONS AND ACRONYMS	xii
ABSTRACT	xiii
CHAPTER ONE.....	1
1. INTRODUCTION.....	1
1.1. Background of the Study	1
1.2. The Motivation for the Study.....	3
1.3. Statement of the Problem.....	4
1.4. Research Questions.....	5
1.5. Objective.....	5
1.5.1. General Objective.....	5
1.5.2. Specific Objectives.....	5
1.6. Significance of the Study	5
1.7. Delimitation of the Study.....	5
1.7.1. Scope of the Study.....	5
1.7.2. Limitations of this Study	6
CHAPTER TWO.....	7
2. LITERATURE REVIEW AND RELATED WORKS	7
2.1. History of Word Sense Disambiguation.....	7
2.2 Basic Concepts of Word Sense Disambiguation and Application Areas	8
2.3. Approaches to Word Sense Disambiguation.....	9
2.3.1 Knowledge-based Approach	9
2.3.2 Machine Learning Approaches.....	10
2.3.3 Hybrid Approaches.....	17
2.4 Different Knowledge Sources of WSD.....	17
2.4.1 Unstructured Resources.....	17
2.4.2 Structured Sources.....	17
2.5 Related Works	18
2.5.1. WSD of Foreign Languages	19

2.5.2. WSD for Ethiopian Languages	20
2.5.3. Summary of Related Works	26
2.6. Overview of the Wolaita Language	26
2.6.1. Morphology of the Language.....	27
CHAPTER THREE	32
2. RESEARCH METHODOLOGY	32
3.1. Data Collection	33
3.1.1 The Main Sources of Data.....	33
3.1.2. Inter-annotation using Fliess kappa.....	34
3.2. Corpus Preparation	35
3.3. Data Preprocessing	36
3.3.1 Data Cleaning.....	36
3.3.2 Tokenization	37
3.3.3 Stop Word Removal	38
3.3.4 Normalization.....	39
3.3.5 Stemming	40
3.4. Feature Extraction.....	41
3.5. Model Selection for the Wolaita Language WSD.....	42
3.6. Model Evaluation Metrics	43
CHAPTER FOUR	44
4. PROPOSED WOLAITA LANGUAGE WSD MODEL	44
4.1. Overview of the System.....	44
4.1.1. Development Tools.....	44
4.1.2. Design Tools.....	44
4.2. The Architecture of the Proposed System.....	46
4.3. Proposed Preprocessing Techniques	47
4.4. Proposed Feature Extraction	47
4.5. Proposed Classifiers for the WSD Model.....	49
4.6. Proposed Evaluation Techniques	51
4.7. Implementation	52
4.7.1 Importing Python Packages and Libraries	52
4.7.2. Preprocessing	53

4.7.3 Implementation of Feature Extraction.....	55
4.7.4 Implementation of Model and Evaluation Metrics	55
4.7.5 Implementation of testing WSD model with unseen data.....	56
CHAPTER FIVE	57
5. RESULT AND DISCUSSION	57
5.1. Overview.....	57
5.2. Result Discussion.....	57
5.2.1. Result set I: Result of Unsupervised Approaches	57
5.2.2. Result Set II: Results of Supervised Approaches	59
5.2.3. Result Set III: Results of Algorithms with Stemming.....	60
5.2.4. Result set IV: The effect of stopwords in algorithms concerning time complexity	62
5.2.5. Result set IV: Average accuracy of selected feature extraction techniques	63
5.2.6 Result set IV: Optimum window size for SVM and NB algorithms.	65
5.2.7. Result set V: Confusion matrix Results in SVC.....	70
5.2.8. Result set VI: Average cross-validation (CV) Results in SVM and NB	72
5.2.9. Statistically significance test for algorithms	73
5.3 Discussion	74
CHAPTER SIX	76
6. CONCLUSION AND RECOMMENDATIONS.....	76
6.1. Conclusion	76
6.2. Recommendation and Future Works.....	77
SPECIAL ACKNOWLEDGMENT	78
REFERENCES	79
APPENDIXES.....	xiv
Appendix I: The selected ambiguous words with their different senses.....	xiv
Appendix II: Sample dataset format	xv
Appendix III: Stopwords of the Wolaita Language	xvi
Appendix IV: Data annotator's result using Fleiss kappa.....	xvii

LIST OF FIGURES

Figure 3. 1 Steps of research methodology	32
Figure 3. 2 Tokenization algorithms (Feyisa, 2015).....	38
Figure 4. 1 Proposed Supervised Approach Architecture of Wolaita Language WSD.....	46
Figure 4. 2 Proposed Preprocessing phase	47
Figure 4. 3 Python Packages and Libraries for our study.....	53
Figure 4. 4 Loading dataset to Python.....	53
Figure 4. 5 Putting data using windowing.....	53
Figure 4. 6 Tokenization implementation.....	53
Figure 4. 7 Stopword removal implementation.....	54
Figure 4. 8 Normalization implementation	54
Figure 4. 9 Implementation of feature extractions	55
Figure 4. 10 Implementation of models and evaluation metrics	55
Figure 4. 11 Implementation of testing model.....	56
Figure 5. 1 Average accuracy for clustering algorithms	58
Figure 5. 2 Average Accuracy of Supervised Algorithms	60
Figure 5. 3 Average Accuracy of supervised algorithms with stemming	61
Figure 5. 4 Average window size accuracy in SVM.....	66
Figure 5. 5 Average window size by computational time in SVM.....	67
Figure 5. 6 Average window size accuracy in NB.....	68
Figure 5. 7 Evaluation metrics average result in SVM and NB	70
Figure 5. 8 Confusion matrix result in train and test data for the 'Haytta' dataset.....	71
Figure 5. 9 Confusion matrix result for the 'Sintta' dataset	72
Figure 5. 10 Average Cross-validation result in SVM and NB	73

LIST OF TABLES

Table 2. 1 Summary of the related works	25
Table 2. 2 Personal pronouns of the language (Mara, 2018)	27
Table 2. 3 Wolaytta Consonantal Phoneme Inventory (Habtamu, 2010)	29
Table 3. 1 The overall data for datasets	34
Table 4. 1 Hardware tools	44
Table 4. 2 Software tools	45
Table 4. 3 Python packages	45
Table 5. 1 Result of Unsupervised approach	58
Table 5. 2 The accuracy result of supervised algorithms.....	59
Table 5. 3 Results with Stemming	61
Table 5. 4 The effect of stopwords concerning time complexity.....	62
Table 5. 5 Average accuracy of feature extraction techniques.....	64
Table 5. 6 Window size experimentation result using SVM algorithm.....	65
Table 5. 7 Window size result by computation time in SVM.....	67
Table 5. 8 Window size experimentation result using NB algorithm.....	68
Table 5. 9 Selected evaluation metrics with SVM.....	69
Table 5. 10 Selected evaluation metrics with NB.....	69
Table 5. 11 Average CV result in SVM and NB	72
Table 5. 12 Probability value (p-value) of five algorithms with SVM.....	74

ABBREVIATIONS AND ACRONYMS

AI	Artificial Intelligence
NLP	Natural Language Processing
WSD	Word Sense Disambiguation
RNN	Recurrent Neural Network
IDF	Inverse Document frequency
POS	Part of Speech
MT	Machine Translation
WLLD	Wolaita Language and Literature Department
FBC	Fana Broadcasting Corporation
EM	Expectation Maximization
NN	Neural Network
NB	Naïve Bayes
RFC	Random Forest Classifier
SVM	Support Vector Machine
SVC	Support Vector Classifier
SGD	Stochastic Gradient Descent
TF	Term Frequency
MLP	Multi-Layer Perceptron
CV	Cross-Validation
DT	Decision Tree
GD	Gradient Descent
SP	Speech Processing
WS	Window size
WSU	Wolaita Sodo University

ABSTRACT

Words that can signify different things in various circumstances are present in all human languages. The term "word sense" in natural language processing (NLP) refers to the various interpretations or meanings that a word may have depending on the context in which it is used. Word Sense Disambiguation (WSD), in the context of natural language processing, has been defined as, a task that involves determining the correct meaning of a word within a given context. Word ambiguity problems have faced the difficulties for Natural Language Processing and computers can't understand ambiguous words as human beings. As the solution to this big challenge, WSD is developed for different languages by different researchers. In the Wolaita language also there are different ambiguous words like in all other languages. So, this thesis presents a research work on Word Sense Disambiguation in the Wolaita Language. To conduct this study, we selected a corpus-based machine-learning approach for 3560 sentences collected from different data sources in the language. To conduct the research, we selected seven ambiguous words from the language namely "Sintta", "Haytta", "Ayfiya", "Doona", "Aadhdha", "Naaga", and "Ogiya" and seven different datasets are prepared. After the dataset was prepared, we applied preprocessing techniques like tokenization, stopword removal, stemming, and normalization. We used BOW, Word2vec, and Tf-idf integrating with N-gram for feature extraction. We tested four different clustering algorithms (EM, simple k-means, farthest first, and hierarchical clustering) for unlabeled data for comparison and we also selected six different algorithms (SVM, NB, NN, Adaboost, RFC, and Bagging) for the supervised approach. Finally, we compared the performance of the algorithms in both clustering and classification models. From the selected clustering algorithms, the EM had the best performance with 63.4%, and from the selected six supervised algorithms the SVM and NB achieved good performance with an accuracy of 86.5% and 84.1% respectively with optimum window size 5-5 for the Wolaita language WSD.

Keywords: *Word Sense Disambiguation, Natural Language Processing, Wolaita language, Machine learning*

CHAPTER ONE

1. INTRODUCTION

1.1. Background of the Study

Nowadays, NLP has expanded its applications to include Machine Translation, Speech Recognition, Information Retrieval (IR), and Information Extraction (IE), and it is now processing vast amounts of natural language data. In natural language processing (NLP), the term "word sense" refers to the different meanings or interpretations that a word can have depending on the context in which it is used. Many words in a language have multiple senses, and understanding the correct sense of a word is crucial for accurate language understanding and processing. Word ambiguity is the challenge in natural language processing of recognizing which sense (meaning) of a word is activated by the use of the word in a given context, a process that appears to be mostly unconscious in people (Navigli., 2009). By facilitating the efficient processing of text and speech, NLP seeks to enhance both human-human and human-computer communication (Jurafsky, 2007).

Word Sense Disambiguation is a natural language processing (NLP) task that seeks to discern the correct meaning of a word in a given context, especially when the word has several interpretations or senses. Ambiguity in a word's syntax or meaning is one of the key problems that any NLP system runs against. By using very precise part-of-speech (POS) taggers, Word's syntactic ambiguity can be eliminated. WSD, on the other hand, relates to the resolution of semantic ambiguity. The resolution of semantic ambiguity is more challenging than that of syntactic ambiguity. However, no application of natural language processing will ever require a computer to recognize and resolve the use of ambiguous words. Human language is ambiguous by nature. All-natural languages are characterized by word sense ambiguity in particular, with many terms in each language having many meanings.

Wolaita is one of the languages in the Omotic family with the biggest native-speaker population among Ethiopia's more than 100 million people. The language is used as the official language in Ethiopia's Wolaita Zone. According to ethnology statistics, 5% to 25% of the population are literate in this language. In Ethiopia now there are more than 6 (six) million language speakers

of this language are live (“Toophphiya Hayyo Billaa Woshettaa,” 2008). As in other languages, there are polysemous words in the Wolaita language, which can make different ambiguities. Semantic lexical ambiguity, which occurs at the word level and can have several meanings depending on the context, is the most difficult problem for NLP among these ambiguities (NG’ANG’A, 2005).

For example, the word ‘**hayttaa**’ has more than one meaning according to the contexts below.

- 🌈 ‘Tana **hayttay** sakes.’ (‘My **ears** are hurting)
- 🌈 ‘Toophphiyassi maratooniyan lo” o **hayttay** de’ees.’ (‘Ethiopia has a good **reputation** in marathons’)
- 🌈 ‘Mitta **hayttay** bone carkkoy carkkiyode qoqofettees.’ (‘The **leaves** of trees scatter in summer when the wind blows hard)

The meaning for ‘**hayttaa**’ in the first sentence is ‘**the ear**’, in the second sentence is ‘**the reputation**’, and in the third sentence is ‘**the leaf**’.

There are three main techniques for developing word sense disambiguation. Knowledge-based, corpus-based, and hybrid techniques are among them. External sources are used for otology information in knowledge-based systems. Machine learning approaches using corpora are subdivided into supervised, semi-supervised, and unsupervised machine learning algorithms. In contrast to unsupervised learning paradigms, which aim for the clustering of unlabeled data based on data similarity, supervised learning is a classification methodology that requires preset human-annotated data with a class. Although both the supervised and unsupervised learning paradigms have constraints that make providing accurate classification and clustering results difficult, these drawbacks are complementary. The training data must be labeled using the supervised technique, which is tedious (expensive), subjective, and time-consuming (Teklay, 2018). In contrast, the unsupervised approach results in substantially lower accuracy producing outcomes that are unsatisfactory for many applications and frequently lead to word meaning sets that are not naturally intuitive to humans because there is no human involvement in label annotation. Between unsupervised learning and supervised learning, semi-supervised learning uses both a sizable amount of unlabeled data and a modest amount of labeled data for training (Ide & V6ronis, 1998). Corpus-based methods and knowledge-based approaches are combined to create a hybrid approach (Segid, 2015). As stated by (Tesema, 2017), using the ideal window

sizes which correspond to the presence of words from the left and right of the ambiguous word improves the effectiveness of corpus-based techniques. The best window sizes differ by language, as many researchers assert. However, as (Pinar Y, 2005) stated, two criteria were used to identify optimal window sizes: the model's accuracy and computing time cost. As we can see from earlier studies, the best window sizes were chosen solely based on average accuracy (Aschale,2020) (Wassie et al., 2014).

In this study, we conducted the efficient WSD of the Wolaita language by comparing different algorithms of machine learning. Our study specifically focuses on the supervised approach and unsupervised, as we discussed below in the literature different local language researchers used supervised, unsupervised, and semi-supervised approaches, but most of them identify only two senses from ambiguous words in the sentences. But in this study, we selected the word with five senses.

Only one study on WSD was undertaken for the Wolaita language (Temesgen, 2021), but the language has several polysemy words. In his research paper, the researcher tried to conduct the WSD by using four machine-learning algorithms for five ambiguous words, as we mentioned above, but in this study, we conducted six algorithms (Naïve Bayes, Support Vector Machine, Neural Network, AdaBoost, Bagging, and Random Forest) for seven ambiguous words. We also conducted another corpus-based approach which is an unsupervised approach and we tried to compare four algorithms from this approach (Simple k-means, Expectation maximization, Farthest First, and Hierarchical clustering).

Because of the relevance of WSD in understanding semantics and numerous real-world applications, researchers have been working hard to solve the challenge. Therefore, this study aims to develop an improved WSD for the Wolaita language by using labeled and unlabeled data.

1.2. The Motivation for the Study

Applications for natural language processing are actively being created for the Wolaita language. The WSD task is a potential intermediary task for many different NLP systems, such as information extraction, machine translation, monolingual and multilingual information retrieval, and natural language understanding. For many Natural Language Understanding applications, it is crucial to resolve word ambiguity. However, the Wolaita language text is

difficult to automatically parse because of the language's high quantity of polysemous words. This encouraged us to create an automatic Wolaita language word sense disambiguation system.

1.3. Statement of the Problem

Wolaita language is one of the major languages mainly spoken in Ethiopia. In the Wolaita language, some words have the same form but have more than one meaning, like in other natural languages. So, it is difficult to understand the meaning of those words in the sentence context. Hence, to clearly understand a word in the language, WSD for the Wolaita language is needed

The Wolaita language has begun to employ technology for a variety of purposes, just like other languages. Wolaita language documents come in many different forms and can be found in many repositories. As was previously said, word sense disambiguation has a variety of applications. The use of WSD in MT, IR, SP, text processing, IE, grammatical analysis, content analysis, and thematic analysis is by far the most widespread. It assists them in overcoming the ambiguity problem.

To solve the problem of ambiguity, many researchers conducted WSD for foreign languages like English, Arabic, and Nepali and also for local languages like Geez, Amharic, Afaan-Oromo, and Tigrigna languages to interact WSD with other NLP applications (Amlakie, 2020; Feyisa, 2015; M. Solomon, 2010). Particularly, in the Wolaita language single word can have numerous meanings, and each can be translated into numerous terms in the target language. Choosing the right definition of a term that best fits the context is a difficult challenge (Dorr, 1994). Future attempts to make computers understand the Wolaita Language are limited by the lack of WSD in the natural language processing system for that language. Temesgen Tadesse (Temesgen, 2021) carried out one study for the Wolaita language WSD. In this research, the researcher tried to disambiguate five ambiguous words using the supervised machine learning approach, but the researcher didn't state how to disambiguate the words if the data is unlabeled as the unsupervised approaches were done without human intervention. So, the researcher only analyzed the potential of supervised machine learning algorithms. With this in mind, we tried to develop an improved WSD for the Wolaita language that has more than five words in this study and the senses of the word are more than the words which are selected in most of the research done before in different languages.

1.4. Research Questions

This research is intended to answer the following questions: -

1. Which machine learning algorithm is most effective for WSD model development?
2. Which features most contribute to enhancing the existing WSD model?
3. To what extent does stemming affect the performance of the WSD model?

1.5. Objective

1.5.1. General Objective

- ✚ The general objective of this study is to compare different machine learning algorithms for resolving ambiguities in the Wolaita language.

1.5.2. Specific Objectives

This study specifically concerns the following specific objectives: -

- ✚ To identify ambiguity in sentences and understand WSD issues in the language.
- ✚ To review and show the important feature extraction technique of the WSD model
- ✚ To investigate the effect of different features on the model's performance
- ✚ To investigate the effect of stemming in WSD model development.
- ✚ To analyze and select effective algorithms for the WSD in the Wolaita language

1.6. Significance of the Study

Incorporating the WSD model for the Wolaita language with the NLP applications provide relevance to society and institutions. This research's main contribution is finding an efficient approach to Wolaita language WSD. Therefore, it will have a significant contribution to the computer in understanding the Wolaita language. The output of this research work will also be a vital component for future researchers and the development of other NLP applications like machine translation, information retrieval, grammatical analysis, question answering, information extraction, content, etc.

1.7. Delimitation of the Study

1.7.1. Scope of the Study

The study focuses on comparing and identifying machine learning algorithms in the Wolaita language. In the Wolaita language, the problem of ambiguity happens at any level like

phonology, morphology, syntax, semantics, and discourse. This study is done for only sentence-level data collected from different sources. The study was only able to train and evaluate utilizing 3560 collected words with seven ambiguous words, due to the lack of a word context database for the Wolaita language. In this study, ambiguous words with five, four, and three meanings were used and in total, seven words had 24 senses. As our study is a comparative analysis of machine learning algorithms in WSD, we selected four clustering algorithms EM, K-means, Farthest First, and Hierarchical Clustering from the unsupervised approach and we selected six algorithms Naïve Bayes, Neural Network, SVM, Random Forest, AdaBoost, and Bagging from supervised learning approach.

1.7.2. Limitations of this Study

Only four clustering and six algorithms for classification bootstrapping were utilized to develop and assess the WSD model due to time limitations on training, testing, and outcomes analysis. The study was only able to test out seven ambiguous words due to a lack of sense-annotated data and linguistic resources. We solely worked with text-based data. Without any type of grammar or spelling correction, the system only works with semantic-level analysis and prediction. In addition to this, we didn't develop the prototype of the model with time constraints and as it was developed by the prior researcher for the Wolaita language WSD.

CHAPTER TWO

2. LITERATURE REVIEW AND RELATED WORKS

This section discusses Word sense disambiguation and its related literature in detail. This chapter also covers the application, the background history, the tasks, and the major approaches that are employed by the WSD researcher are discussed. Additionally, various machine learning methods that will be used in this research as well as other algorithms that have been demonstrated to perform well in WSD research are presented.

2.1. History of Word Sense Disambiguation

In the early stages of machine translation, WSD was initially developed as a separate computing activity in the 1940s. Early academics claimed that a computer could not be programmed to determine the meaning of a word. Considering how difficult it was at the time (Yehoshua Bar-Hillel, 2021). As a result, it ranks among the earliest computational linguistics issues. Warren Weaver first presented the idea in a computational framework in 1949. The semantic interpretation systems developed within the artificial intelligence field until the 1970s included WSD as a subtask. However, WSD systems were vulnerable to a knowledge acquisition bottleneck because they were mostly rule-based and hand-coded at the time (Yarowsky, 1992).

Knowledge-based strategies first came into being in the 1980s with the publication of the Oxford Advanced Learner's Dictionary. Because of this, knowledge-based techniques were employed to address the hand-coding issue. But dictionaries were employed during the disambiguation procedure. (Yehoshua Bar-Hillel, 2021).

In the 1990s, WSD was advanced by the growth of computational linguistics and statistical machine learning. To deal with ambiguity, the three machine learning algorithms—supervised, unsupervised, and semi-supervised learning algorithms—are utilized. However, until they reach a plateau, supervised learning algorithms perform better than alternative algorithms. (NG'ANG'A, 2005) (Tadesse Tilahun, 2021). Since supervised techniques' accuracy peaked in the 2000s, the focus has switched to coarser-grained senses, domain adaptability, semi-supervised and unsupervised corpus-based systems, combinations of diverse methods, and the resurgence of knowledge-based systems through graph-based techniques. The best performance is still reserved for supervised systems (Teklay, 2018).

WSD research has been done in Ethiopia for five different languages: Amharic, Afan Oromo, Tigrigna, Geez, and, most recently, Wolaita. The three approaches corpus-based approaches, knowledge-based approaches, and hybrid approaches are used to decipher the meaning of the ambiguous phrase.

The corpus-based approach was the first conducted approach for the Amharic language among the approaches by Teshome (Teshome, 2009). He used semantic vector analysis to illustrate WSD. Later, several techniques were used by other researchers to enhance his work and promote the WSD of Amharic. As a result, the WSD of the Amharic language was carried out using hybrid, knowledge-based, and corpus-based approaches (Wassie and Solomon, 2014) (Segid, 2015).

2.2 Basic Concepts of Word Sense Disambiguation and Application Areas

Word sense is a definition or meaning of a word (Mante & Kshirsagar, 2014) (Tesema, 2016) and Disambiguation means to eliminate all ambiguity. In natural language, many words have multiple meanings, and each one is influenced by their context. The process used to identify the meaning of the word from the context is known as word sense disambiguation. WSD is a method for determining a word's meaning within a phrase. A word may have several meanings, and humans determine the precise meaning of a word based on context. Similar techniques can be used by computers to determine the meaning of unclear words in sentences. The technique of identification is also used to determine the proper meaning of an ambiguous word in a particular situation (Amlakie, 2020). Any WSD task seeks to make it possible for machines to recognize these ambiguous words' accurate meanings like that of humans.

In WSD tasks, there are always two steps, according to Ide & Vronis' (1998) study. Making a list of all the uses of each word that are (at the very least) pertinent to the text or discourse in question is the first stage. Select a list of senses from a common dictionary, a list of synonyms from a thesaurus, or a list of translations from a translation dictionary to complete a sense inventory. The second phase entails a method for accurately and swiftly associating each time a word appears with its proper meaning. In the realm of natural language, the question of how to define a sense correctly is still up for debate (Ide & Vronis, 1998).

Text processing, speech recognition, information retrieval, machine translation, and other applications all require word sense disambiguation. Data processing is error-prone without WSD

(Majumder, 2021). Word sense disambiguation has the potential to improve NLP when used correctly. Utilizing WSD can significantly increase a system's output's accuracy. It preserves the word's meaning given the context.

2.3. Approaches to Word Sense Disambiguation

Recently, several methods are employed to handle Word Sense Disambiguation difficulties based on the knowledge type that is used. Contextual embedding without Different methods was employed by WordNet to address WSD issues in various languages throughout the world. Each of these three approaches, employs various protocols, knowledge bases, and algorithms. The majority of disambiguation techniques tend to concentrate on finding contextual clues that are unique to a term and can be utilized to differentiate between its senses. The difficulty of acquiring these hints or signs has been attributed to the significant human participation required for each word, which results in a vocabulary coverage issue. In WSD literature, this is referred to as the knowledge acquisition bottleneck. There are three different WSD methodologies: knowledge-based, corpus-based, and hybrid, which combine knowledge-based and corpus-based methodologies.

2.3.1 Knowledge-based Approach

This approach is also mainly known as the Dictionary-based approach. Knowledge-based approaches draw on outside information sources that provide explicit sense distinctions to determine the appropriate sense of a word in context. The purpose of the knowledge-based method WSD is to use knowledge resources to infer the senses of words in context. Large-scale lexical resources including dictionaries, thesauri, and corpora became widely accessible in the 1980s and 1990s, which marked a turning point in the development of WSD. These strategies primarily aim to eliminate the requirement for a lot of training material by utilizing the information found in dictionaries (Dureti, 2017).

The knowledge-based approach aims to heavily rely on knowledge sources for figuring out what words represent in a certain context. Although knowledge-based approaches were found to be less effective, alternative supervised procedures also have several other advantages. Thesauri, dictionaries, and collocations are the most often used resources in this strategy. (Borah et al., 2014). The benefit of using knowledge-based approaches over supervised learning and clustering is that training data are not needed for every word that needs to be disambiguated.

This enables the system to do all-word disambiguation, which is the process of separating words in running text. Some of the dictionary-based approaches are discussed below:

Over-lap-based approach: - A machine-readable dictionary technique (MDR) is necessary for the overlap-based approaches (Borah et al., 2014). It involves figuring out the various aspects of the confusing word senses as well as aspects of the word in context. The most overlapped word sense is chosen as the one that best fits the context.

Selection preferences: - Selection preferences are limitations on the semantic type that a word sense places on the words it links with, typically through grammatical relationships in phrases. They frequently have WSD-relevant information. Their application is constrained, and more sources of information are needed for a complete and accurate WSD. One method of limiting the potential meanings of a word in a particular context is to use selection preferences in some of the earliest algorithms for word sense disambiguation. Selection preferences are a representation of common-sense knowledge about classes of concepts as well as information about the potential relationships between word categories. The mainly used algorithms used in the selection preferences algorithm are (Segid, 2015):

2.3.2 Machine Learning Approaches

These approaches are commonly known as corpus-based approaches, as they use the manually prepared corpus to train the algorithms. Using a corpus-based technique, word senses are extracted from a vast set of annotated data that has been sense-tagged. A classification model is constructed using examples in a corpus-based manner. These techniques have two phases: learning and classification. The learning phase entails learning a sensory classification model from the training examples. To allocate the output senses, the categorization procedure applies to fresh examples. The presentation of the examples using features or attributes is one of the first and most crucial considerations (Segid, 2015). Machine learning or corpus-based approaches mainly have three different approaches namely: Supervised approaches, Semi-supervised approaches, and Unsupervised approaches.

2.3.2.1 Supervised Corpus-based Approaches

The NLP community has seen a dramatic transition over the past 15 years from the usage of manually created systems to the use of automated classification techniques. A classifier is generated from manually sense-annotated data sets using the automatic classification method

known as supervised WSD. These approaches take the data that are manually annotated. The necessary concepts from a sense inventory are manually annotated onto the occurrences in the training data. A supervised learning algorithm learns to recognize the context around these concepts to automatically assign ideas to instances in the test data containing the target word. This process results in a model. Supervised learning techniques generally surpass other WSD techniques in terms of very high disambiguation accuracy (Segid, 2015).

Supervised approaches are predicated on the idea that the context can give sufficient data on its own to distinguish between words, therefore reasoning and general knowledge are not considered necessary. Since these supervised approaches rely on substantial amounts of manually sense-tagged corpora for training, which are time-consuming and expensive to produce, they are subject to a new knowledge acquisition bottleneck.

A supervised technique makes it possible to connect contextual information (world knowledge) to word sense by training the sense model using sense-tagged corpora. It ought to perform better than unsupervised methods because more data is fed into the system. Nowadays, there are more training corpora than ever before, hence most WSD algorithms that have recently been developed are supervised (Zhou & Han, 2005). The supervised approach always employs two sets of data for testing and training and requires tagged corpora as a training set. Even though manually constructing labeled corpora takes money, it is very successful. For WSD, various supervised approaches are employed; the most popular ones are discussed below:

2.3.2.1.1 Learning Algorithms under supervised approach

Naïve Bayes (NB): is one of the most straightforward examples of a probabilistic learning technique. The Naive Bayes Classifier is a supervised machine-learning method that makes use of Bayes' Theorem, which presumes that features are statistically independent. The Bayesian Classification is both a statistical and supervised learning method for classifying data. Calculating the probability of the outcomes assumes an underlying probabilistic model and enables us to convey uncertainty about the model systematically. Thomas Bayes (1702–1761), who came up with the Bayes Theorem, is honored by the name of this algorithm, which helps handle diagnostic and predictive issues. Prior knowledge and observed data can be integrated with Bayesian classification, which offers useful learning techniques (Feyisa, 2015). Naive Bayes classifiers are one of the most known algorithms for learning-to-text classifications.

Decision Tree: in classification and prediction problems, decision trees are a predictive modeling tool employed. To categorize objects into values of the dependent variable depending on the values of independent variables, a classifier is employed in data mining and expressed as a recursive partition of the instance space. Decision trees come in two basic varieties. Both classification and regression trees are being used here. Since they group examples into groups or classes, classification trees are decision trees that are used to forecast categorical variables. As a result, there is a chance that the data will be accurately classified. The class probability, or confidence that a record belongs to a particular class, is reported in this. Regression trees, a type of decision tree used to predict continuous variables that are not nominal, are the second option. It calculates an estimate for a target variable with a numerical value (Loh, 2011) (Feyisa, 2015).

Neural network: artificial neurons are utilized in the neural network-based computational model for data processing utilizing the connectionist method (Mcculloch & Pitts, 1990). The pairs of input characteristics provide the learning program's input, and the purpose is to divide the training context into non-overlapping sets. The newly generated pairs and link weights are then gradually changed to produce a higher activation. Words can be represented as nodes in neural networks, and these words will activate the thoughts to which they are semantically linked. The inputs are transferred from the input layer to the output layer via all intermediate layers. The input can be readily transported through the network and controlled to produce an output. A network with connections that stretch in all directions and generate loops makes it difficult to compute a distinct output. Feed-forward networks are typically a better alternative for non-time-dependent problems that forecast a wide range of applications (Birhane, 2017).

Support vector machine: the principle of structural risk minimization is used by support vector machine-based algorithms (Buscaldi et al., 2006), the purpose of this technique is to differentiate positive examples from negative instances with the greatest possible margin, where the margin is the distance between the positive and negative examples. Support vectors are the positive and negative instances that are closest to the hyperplane. SVM-based algorithms are used to divide a small number of instances into two separate classes. This algorithm finds a hyperplane between these two classes to maximize the separation margin between them. The classification of the test example is determined by which side of the hyperplane the test example is located on. The input features can also be translated into a high dimensional space, however, in that

case, specific kernel functions are utilized to lower the computing cost of the training and testing procedures in high dimensional space. In the case of non-separable training samples, a regularization parameter is utilized. This parameter's default value is deemed to be 1. This regularization method manages the trade-off between a big margin and a low training error (Birhane, 2017).

2.3.2.2 Semi-supervised corpus-based approaches

Semi-supervised learning is a technique that lies between unsupervised learning and supervised learning in that it uses both a small number of labeled and a big amount of unlabeled data for training. Semi-supervised or minimally supervised approaches are gaining popularity because of their ability to perform well on big data sets with only a limited quantity of annotated reference material. There are numerous ways and approaches for learning essential qualities from auxiliary data and clustering or annotating data using the gained knowledge. For each word, the bootstrapping approach begins with a limited number of seed data: either manually tagged training instances or a small number of surefire judgment criteria (e.g., play in the context of bass usually indicates the musical instrument). Using any supervised approach, the seeds are utilized to train an initial classifier. This classifier is then used in the untagged section of the corpus to generate a bigger training set that includes only the most confident classifications. The procedure is repeated, with each new classifier being trained on a larger and larger training corpus, until the entire corpus is consumed, or a specified maximum number of iterations is reached. Bootstrapping (BS) appears to be a supervised technique, but it requires only a few seeds rather than a vast number of training samples (Segid, 2015). In one sense, these algorithms are used when supervised learning is fed increasing amounts of unlabeled data to categorize the unlabeled data into a particular class, and in the other, they are used when an unsupervised algorithm is fed increasing amounts of labeled data to cluster the data (Amlakie, 2020). There are different algorithms in the semi-supervised approach and some of them are discussed below:

ADTree: by utilizing boosting to add the various branches, ADTree (Freund, 1999) constructs what is known as an alternative decision tree. A set of decision stumps with numerical leaves and connections between each leaf and a group of other stumps makes up an alternative decision tree. By adding up all the numerical nodes encountered when traveling the many branches of the tree that are appropriate for the examples, the tree can be used to categorize unlabeled

instances into binary classes. The anticipated type is determined by the sign of this value (Amlakie, 2020).

Bootstrapping: the use of unlabeled data in the learning process with the need for a predetermined sense inventory for target words is how semi-supervised approaches for WSD are distinguished. In WSD, co-training and self-training are the two primary approaches to semi-supervised learning techniques (Wassie et al., 2014). Most semi-supervised learning techniques for WSD employ bootstrapping algorithms. To get around the difficulty of acquiring huge amounts of sense-tagged data, bootstrapping sense-tagged seed samples are employed. It operates by using a model discovered from an enhanced labeled dataset in a previous iteration, iteratively classifying unlabeled examples, and inserting confidently identified examples into labeled datasets. It has been discovered that this bootstrapping procedure does not properly explore the affinity information across unlabeled instances. Bootstrapping is based on the local integrity assumption, which also underlies many supervised learning techniques. instances close to labeled examples within the same class will have the same labels. By creating a sense classifier with little training data, bootstrapping aims to solve the two fundamental supervision issues of sparse data and a lack of annotated data (Amlakie, 2020).

Bagging: Its name signifies bootstrap aggregation. To lower the variance and increase the precision of a particular predictor, the bagging technique employs bootstrap sampling. It applies to both regression and classification. Think of a dataset with a size of N , $Z = z_1, z_2, \dots, z_N$, where $z_i = (x_i, y_i)$, where y_i represents a class label in classification issues or a real value in regression problems. The goal of bagging is to combine a set of B predictors that were previously learned (each from a bootstrap sample $Z_b \subset Z$, for $b = 1, \dots, B$). The majority vote is how this algorithm operates (Ferreira & Figueiredo, n.d.). The bias-variance decomposition's variance component is reduced when many predictors are combined, which lowers the anticipated error. The number of classifiers used in the ensemble has an impact on how much of this variance is reduced. The major factor contributing to the bagging approach's success in classification is the way it enhances classification outcomes if the basic classifiers are unstable. A majority vote on the work produced by the weak learner determines the final categorization (Teklay, 2018).

AdaBoost (AB): is a broad technique for combining numerous weak classifiers, each of which may only be moderately accurate, to produce a highly accurate classification rule. The basic

goal of the AdaBoost algorithm is to create a strong classifier with an arbitrarily low error rate on the training set by linearly combining numerous simple and not always exact classification rules (also known as weak rules or weak hypotheses). AdaBoost has been used to solve a variety of real-world issues, including various NLP problems (Agirre & Martinez, 2001).

2.3.2.3 Unsupervised corpus-based approaches

Unsupervised approaches rely on training data that has not been annotated. These approaches, in general, do word sense discrimination rather than disambiguation. Disambiguation aims to determine the right concept of an instance given a sense inventory, whereas discrimination seeks to cluster instances of a given target word so that instances that utilize the same concept of the target word are in the same cluster. However, the words in a test data set must be disambiguated to evaluate clustering approaches. Clustering has the advantage of not requiring a significant amount of manually annotated training data because the labeling of the instances in the training data is done by clustering algorithms rather than by human annotators as with supervised approaches. Another benefit of clustering is that it is independent of language and domain, requiring simply a corpus in the language and domain of interest. The problem is that training data is necessary for each word to be disambiguated, and this method has historically not achieved as high disambiguation accuracy as supervised methods. Unsupervised approaches offer the ability to overcome the knowledge acquisition bottleneck, namely the scarcity of large-scale resources that have been manually tagged with word meanings (Segid, 2015). They do not employ labeled training text and, in their purest form, do not utilize machine-readable resources such as dictionaries, thesauri, ontologies, etc. The fundamental disadvantage of wholly unsupervised systems is that they cannot rely on a shared reference inventory of senses because no lexicon is used. Some of the unsupervised learning approaches are discussed below:

Context clustering: a context clustering approach is based on clustering methods in which context vectors are first constructed and then clustered to discover the meaning of the word (Birhane, 2017). This approach uses vector space as word space, with dimensions limited to words only. A word in a corpus will also be denoted as a vector in this manner, and the number of times it occurs within its context will be counted. The co-occurrence matrix is then generated, and similarity measurements are applied. Then, using any clustering technique, discrimination is conducted. The offline process employs a distributed K-means clustering method.

Hierarchical clustering: unlike a partitioning technique, a hierarchical clustering method groups data objects into a tree of clusters. We distinguish between divisive and agglomerative clustering depending on whether the clustering is carried out top-down, i.e., from a single cluster to the maximum number of clusters, or bottom-up, i.e. from the maximum number of clusters to a single cluster (Teklay, 2018). Numerous application domains are very interested in hierarchical clustering solutions, which take the shape of trees called dendrograms. A perspective of the data at various levels of abstraction is offered by hierarchical trees. When conducting data analysis, flat partitions of various granularities can be retrieved due to the consistency of clustering solutions at many degrees of granularity, making them perfect for interactive visualization and exploration. Additionally, there are other instances in which clusters have subclusters, and the hierarchical structure is, in fact, a natural constraint on the underlying application area (for example, biological taxonomies, phylogenetic trees, etc.) (Zhao & Karypis, 2005).

K-means clustering: k-means is an iterative, numerical, unsupervised, non-deterministic approach. The method has proven to be quite effective in many real-world applications since it is quick and easy to use, and it can result in good clustering results. But it is excellent for creating globular clusters (Shi et al., 2010). K-means is a partitioning clustering algorithm that divides the given data items into k separate clusters. This method is iterative and converges to a local minimum in most cases. Therefore, the results of formed clusters are numerous and distinct from one another. There are two distinct steps to the algorithm (Teklay, 2018).

(i). The user chooses k centers at random in the first phase, where the value of k is predetermined. to transport every piece of data to the closest hub. The distance between each data object and the cluster centers is calculated using several distance functions. The initial grouping is finished once all of the data objects are contained in a few clusters.

(ii). Following that, the average of the recently formed clusters is recalculated in the second phase. Up until the criterion function is the least, this iterative process keeps going back and forth (Xie & Jiang, 2010).

Farthest First Clustering: this is a variant of K-Means that positions each cluster center at the location furthest away from the existing cluster center. This point must be contained within the data region. In most cases, this significantly boosts clustering speed because less reassignment

and change are required. The fastest-point heuristic-based method is appropriate for large-scale data mining applications (N. Sharma et al., 2012).

Expectation maximization: This is an iterative technique for determining maximum likelihood or Maximum a posteriori (MAP) estimates of parameters in statistical systems that depend on latent variables that are not seen. The EM iteration alternates between performing an expectation (E) step, which creates a function for the expectation of the log-likelihood evaluated using the current estimate for the parameters, and a maximization (M) step, which computes parameters maximizing the expected log-likelihood found on the E step (Bala et al., 2014).

2.3.3 Hybrid Approaches

Both corpora and explicit knowledge bases are used in hybrid techniques to gather information for disambiguation. The goal of hybrid systems is to increase WSD accuracy by utilizing the benefits of both methodologies while overcoming their respective weaknesses. For instance, Yarowsky (Yarowsky, 1992) employed bootstrapping methodologies, where the beginning data originates from an explicit knowledge source and is later improved using data collected from corpora. For each sense of each term, he provides a select few seed definitions. The obvious cases in a corpus are then categorized using the seed definitions.

2.4 Different Knowledge Sources of WSD

A key element of WSD is knowledge. The information that knowledge sources supply is crucial for connecting words and sensations. Machine-readable dictionaries, thesauri, glossaries, ontologies, etc., can range from corpora of texts that are either unlabeled or annotated with word senses (Navigli., 2009). Knowledge sources are classified into structured and unstructured sources, and are discussed below:

2.4.1 Unstructured Resources

A corpus is a collection of documents used for learning language models and is an unstructured knowledge resource. Corpora can be sense-annotated or unannotated (i.e., unlabeled). WSD employs both types of resources, which are most useful in supervised and unsupervised procedures, respectively (Navigli., 2009) (Birhane, 2017).

2.4.2 Structured Sources

Structured resources contain the following sources according to (Navigli., 2009): -

Ontologies: - are conceptualization guidelines for certain topics of interest; they frequently include a taxonomy and a set of semantic relations. Ontologies in this sense include WordNet and its expansions, the Omega Ontology, an attempt to restructure and comprehend WordNet, the Simulation of Urban Mobility (SUMO) higher ontology, and others. (Birhane, 2017).

Machine-readable dictionaries (MRDs): - Since the first dictionaries—the Collins English Dictionary, the Oxford Advanced Learner's Dictionary of Current English, the Oxford Dictionary of English, and the Longman Dictionary of Contemporary English—became accessible in electronic format in the 1980s, MRD has become more and more well-liked as a resource for natural language processing (LDOCE). Initiatives to automatically extract lexical and semantic knowledge bases from MRDs were a significant area of research in the 1980s. Despite providing comprehensive information at the lexical level, dictionaries fall short of providing pragmatic information, which is crucial for determining sense (Dureti, 2017).

Thesauri: - A thesaurus is an organized knowledge source that offers details on word relationships such as synonymy, antonymy, and potentially more. Roget's International Thesaurus, which was made machine-tractable in the 1950s and has been utilized in a range of applications including content analysis, machine translation, and information retrieval, is the thesaurus that is most commonly used in the field of WSD. The thesaurus also offers a clear concept hierarchy with up to eight successively more complex levels. Each time a word appears under a distinct category in a thesaurus, it usually refers to one of the word's various meanings; hence, the categories approximately correlate to word senses. Additionally, some scholars make use of Bernard's 1986 Macquarie Thesaurus, which contains more than 200,000 synonyms (Dureti, 2017).

2.5 Related Works

For WSD, a significant amount of research has been conducted in English, Chinese, European, and other languages. The majority of foreign languages provide sufficient tools for NLP tasks. As a result, several sorts of research have been carried out to address a variety of NLP issues using knowledge-based methods that incorporate graph-based models and deep-learning algorithms (Faisal et al., 2018). Only Tigrigna, Amharic, Afaan Oromo, geez, and one research were done in the Wolaita language in Ethiopia, where there are 46–85 spoken languages. Due to the creation of WordNet, knowledge-based methods have recently been used in the Afaan

Oromo and Amharic languages (Temesgen, 2021). For the Wolaita language, only one research was conducted (Temesgen, 2021), he used corpus-based approaches to address the WSD problem of the language. He conducted the WSD by using only the bootstrapping algorithms.

2.5.1. WSD of Foreign Languages

2.5.1.1. WSD for the English language

The Lesk algorithm, a type of machine learning, was created by Lesk Michael (Lesk, 1986) in 1986. The algorithm is used to distinguish between the senses of words with multiple meanings and to overlap word definitions from the Oxford Advanced Learner's Dictionary of Current English (OALD). He built the algorithm based on two presumptions.

First, if it seems as though each term may be used to discuss the same subject, then the dictionary definition must include some everyday words.

Secondly, if two words are used close together in a sentence, they must be discussing the same subject. The algorithm can disambiguate neighboring words in a sentence by comparing their definitions and picking those senses whose definitions have the greatest number of common words. The biggest drawback of this algorithm is that dictionary definitions are often very short and just do not have enough words for this algorithm to work well.

The adoption of Lesk's dictionary-based WSD method was proposed by Banerjee and Pedersen (Banerjee & Pedersen, 2002) says, instead of referencing a regular dictionary as their source, they created the lexical database WordNet for their strategy. This expands comparisons to include the glosses of words related to the words being disambiguated, whereas the original Lesk algorithm improves discovering overlaps in the gloss of nearby terms. They found that the English SENSEVAL-2 lexical sample was evaluated using this method with an accuracy of 32%, which is twice as accurate as the conventional Lesk algorithm.

2.5.1.2. WSD for Indian words

Rohana Sharma and Parteek Bhatia (R. Sharma & Bhatia, 2008), carried out the WSD for Indian languages. The authors make use of a range of word sense disambiguation (WSD) methods, including knowledge-based, machine learning-based, and hybrid strategies. To address the issue of word sense disambiguation, this study makes use of the Hindi WordNet created by the IIT, which comprises a variety of terms and their sets of synonyms, or synsets. By comparing the

multiple definitions of terms in the phrase with the word contained in the synset form of WordNet and the information linked with these words in the form of parts of speech, the researcher attempted to clarify the ambiguity of utilizing the words in these synsets.

Finally, the researcher discovered that knowledge-based approaches are the best strategy for word sense disambiguation, and Lesk's algorithm was used as an illustration to demonstrate its applicability for Hindi WSD. The author took a sample paragraph, constructed its context bag, extracted the semantic bag for the word to be disambiguated, and then used the overlap between the two bags to correlate to each sense of the word to determine which sense of the word is acceptable. The built Hindi WSD system's performance is not, however, assessed by the researcher using a knowledge-based approach.

2.5.1.3. WSD for Bahrain language

Using supervised learning algorithms, Edil Faisal (Faisal et al., 2018) carried out WSD for Bahasa Indonesia. To clarify the meaning of two unclear words "motor" and "bulan," the researchers in this study obtained information from a Wikipedia article.

For feature selection and classification methods like SVM, multivariate Bournoli nave Bayes, and Multinomial Naive Bayes, they employed the TF-IDF and SVD algorithms. The highest accuracy was finally reached using TF-IDF with SVM. However, only two ambiguous words with similar senses were found in this study.

2.5.2. WSD for Ethiopian Languages

Getahun Wassie (Wassie et al., 2014), created an Amharic WSD prototype utilizing a semi-supervised learning technique. Among the 1031 datasets, they only chose five ambiguous words (atena, derese, tenesa, Bela, and ale) for the sample. As supervised learning calls for a lot of human involvement, it falls under the category of training. Three semi-supervised algorithms—Adaboost, Bagging, and ADTree—were utilized in the study. The window size was 3-3, and the methods' accuracy was 84.9%, 81.25%, and 88.45%, respectively. ADTree scored the highest in their study. Their study was limited by the fact that their model only distinguishes between two senses and five words; they chose the size of the window based on average accuracy, but it must also take into account processing time and the nature of meaning, particularly semantic meaning.

Solomon Makonnen (M. Solomon, 2010) focuses on the issue of automatically determining a word's correct meaning in uncertain situations. Additionally, they used a dictionary containing 1045 examples of texts to translate five uncertain words from the British National Corpus (BNC) into Amharic: **(Tena)**, **(mesal)**, **(me'sa'sat)**, **(metrat)**, and **(qereSe)**. They also came to the following conclusions: k means, EM and agglomerative single, average and complete link, and Nave Bayes are the five clustering techniques. They obtained accuracy for those algorithms in the range of 70% to 85%, with the Naive Bayes algorithm performing best overall.

Segid Hassen (Segid, 2015), is another scholar who has used WordNet to disambiguate Amharic word senses. WordNet was one of the knowledge-based WSD techniques he used. The researcher created 10,000 synsets and 2000 words in Amharic WordNets. In his research, he used SQL server, Java, and Python computer languages as development tools. The SQL server was utilized to produce Amharic WordNet, while the Java and Python programming languages were employed to create the study's prototype. The Python programming language was also used to write the root word extraction. He conducted two tests to evaluate the efficacy of the proposed system: identifying the ideal context window size and analyzing the influence of the morphological analyzer.

In Amharic WordNet, he attained an accuracy of 80% with a morphological analyzer and 57.5% without. For some ambiguous terms, the second experiment tested 1-1 windows up to 5-5 windows on both sides of the target word. After the second experiment, the researcher compared the data for each window size and attained superior accuracy at two-two window sizes, 86.5%. As a result of his research, the two-two window size was determined to be the best suitable for the Amharic language.

Solomon Assemu (A. Solomon, 2021), also used a corpus-based strategy to do research on WSD for Amharic. He used an unsupervised strategy to determine the correct sense of ambiguous words using corpus-based approaches. He looked into his experiment using five chosen ambiguous Amharic terms that were also used in (M. Solomon, 2010). He obtained the 1045 sense examples of ambiguous terms for his corpus from Solomon (M. Solomon, 2010), which suggests that they were used in his experiment. When sufficient sense examples are available, the researcher tries to use a balanced distribution of the ambiguous terms' senses to enhance the effectiveness of his study. He performed preprocessing chores on those sense instances before

utilizing them in his experiment to ensure they were suitable for training and assessing using the chosen algorithms.

It was not necessary to separate the corpus into training sets and testing sets for evaluation purposes because the researcher used an unsupervised approach for his research. Here, annotated datasets were employed for the evaluation of the clustering assignment techniques rather than for clustering the meaning of the ambiguous term. The researcher chose five clustering methods for experimentation in his study, including K-means, Agglomerative single, Average and full link clustering, and Expectation Maximization. The Weka 3.6.4 package was the tool he utilized to carry out his experiment. Except for the methodologies he employed, the researcher used the same experimental configurations and data sets as those described in (M. Solomon, 2010). He made an effort to use the same experimental settings and data sets primarily to compare the outcomes produced by supervised and unsupervised procedures. Using the five clustering techniques he chose; he ran four tests in his study. The experiments examined the impact of stemming on the accuracy of the result, the best context window to use, the distribution of training data's impact on the accuracy of the result, and the comparison of the results obtained using the various clustering algorithms used in his study with those obtained using the supervised algorithm, the Naive Bayes classifier.

For five of the chosen Amharic ambiguous terms, he attained accuracy in the range of 65.1% to 79.4% for basic k means, 67.9% to 76.9% for EM, and 54.4% to 71.1% for complete link clustering algorithms. The advantage of this research is that it discriminates between various groups based on the context of polysemous words without explicitly identifying them. The drawback of this research is that unsupervised methods cannot rely on a shared reference inventory of sense because training data is needed for each word that needs to be disambiguated. The researcher's method is based on the idea that words with the same meaning would typically have similar neighbors. It also involves grouping words occurrences and classifying newly occurring occurrences into induced clusters.

Teklay Muruts (Teklay, 2018), used the semi-supervised machine learning approach in an attempt to create the WSD prototype model for the Tigrigna language. They also looked at five ambiguous Tigrigna words, including **kefele**, **OareQe**, **seOare**, **Halefe**, and **medeb**, from 1250 sentences that had been collected.

For grouping the sentences into their different senses, we used four clustering techniques (EM, Simple K-Means, FarthestFirst, and Hierarchical Clustered), and five classification algorithms (ADTree, AdaBoostM1, Bagging, SMO, and Nave Bayes). They also employed WEKA 3.8 because it has the aforementioned algorithms. And they discovered that the semi-supervised strategy outperformed the three machine learning approaches. They used the ADTree, SMO, AdaBoostM1, Bagging, and Naive Bayes algorithms to produce 93.6636%, 91.9224%, 85.35%, 79.1917%, and 70.8968% from their model, respectively. They used the window size of 1-1, this smaller window sizes can also make sense to be syntactic rather than semantic meaning.

Fayisa G. (Feyisa, 2015), Unsupervised machine learning methods are used to analyze a corpus of the Afaan Oromo language in the corpus-based approach to disambiguation that is given. In the current version of the Weka 3.6.11 package, he investigated five clustering methods (basic k means, hierarchical agglomerative: Single, Average, full link, and Expectation Maximization algorithms). The preprocessed dataset was used to train the chosen algorithms using the "Cluster via classification" evaluation method. For the seven ambiguous words **sanyii**, **karaa**, **horii**, **sirna qophii**, **ulfina**, and **ifa**, a total of 1500 Afaan Oromo sense instances were gathered.

The sense example sentences were subjected to various preprocessing procedures such as tokenization, stop word removal, and stemming to prepare them for experimentation. A common strategy for WSD is to consider the context of the ambiguous word and the use of the data from its nearby or collocated terms. A training dataset was applied using a common performance evaluation matrix to assess the system. Three of the five clustering methods that were chosen for the experiment had the highest accuracy ratings. They are Expectation Maximization, which has an average accuracy of 78.9%, Complete Linkage, which has an average accuracy of 76.1%, and Simple K Means, which has an average accuracy of 81.9%.

Tesfa K. (Tesfa, 2013), tried to solve the problem of word sense disambiguation (WSD) for the Afaan Oromo language. To automatically obtain disambiguation information, he developed a corpus-based method for disambiguation in which supervised machine learning techniques were used to a corpus of the Afaan Oromo language.

For his experiment, he used the Naive Bayes theorem to determine the sense's prior probability and likelihood ratio in the relevant context. The system assigns sensations to unobserved examples using data from the training corpus. Therefore, he started from scratch when creating

the Afaan Oromo corpus. The corpus contains 1240 sentences and he evaluated 5 Afaan Oromo ambiguous words namely **sirna**, **karaa**, **sanyii**, **qophii**, and **horii**. He ran two tests utilizing these phrases, the first of which was a performance assessment of the algorithm. This experiment employs a 10-fold cross-validation assessment technique to assess the algorithm's performance. Finally, utilizing the test set's sensible examples, the actual test was run. Nine folds were utilized to train the developed system during this procedure, and the final tenth fold was used to test the system that had been trained on the earlier nine folds. The procedure was done ten times, using nine of them as practice runs and the tenth as a test. The system was evaluated using an average of 124 Afaan Oromo sentences following each training period. Finally, he concluded that a four-word window on either side of the ambiguous word is sufficient for Afaan Oromo WSD in the second trial. He had 79% accuracy in the first experiment.

Temesgen T. (Tadesse, 2021.), tried to solve the problem of WSD for the Wolaita language. He used a corpus-based approach to disambiguation where supervised machine-learning techniques are applied to a corpus of the Wolaita language to acquire disambiguation information automatically. Hence, they developed the Wolaita language corpus from scratch. The corpus contains 2797 sentences and he evaluated five ambiguous Wolaita words named: **‘Doona’**, **‘ayfiya’**, **‘naaga’**, **‘adhdha’**, and **‘ogiya’**. For the ap, He used SVC, Bagging, Random Forest Classifier, and AdaBoost classifier for the approach four algorithms, SVC, and Bagging classifier achieved higher accuracy, 83.22% and 82.82 respectively. They chose WS11(5-5) by using 10-fold cross-validation. In addition to this, they experimented with the base estimator of the bagging classifier to improve the performance of the bagging model. Linear SVC with a Bagging classifier achieved higher accuracy. Therefore, instead of using a DT base estimator, using the best weak learner can increase the performance of the Bagging classifier in terms of accuracy.

Amlake A. (Aschale, 2020), carried out WSD using corpus for the Geez language. His research examined the average accuracy of three corpus-based approaches—supervised, unsupervised, and semi-supervised. SMO, Naive Bayes, and bagging were the methods he employed for the supervised learning approach. He employed EM, Simple K-means, Farthest First, and Hierarchical clustering for the unsupervised learning strategy. Additionally, he employed the AdaBoost and ADTree algorithms for semi-supervised learning. Semi-supervised learning

techniques, particularly ADTree, attained an average accuracy of 91.1% among the three methods. He used 2119 sense instances for six ambiguous phrases to train the machine learning algorithms. Window size 4 was employed to determine the precise meaning of the uncertain word from the surrounding context. As in earlier corpus-based research, the information bottleneck is this study's weakness. In addition, he only used two senses for six ambiguous terms, even though a word may have more than two senses (Temesgen, 2021).

Table 2. 1 Summary of the related works

Authors and year	Objectives	Algorithms used	Gaps
Getahun Wassie (2014)	To find the sense of Amharic ambiguous words using semi-supervised learning.	AD Tree on the window size (3-3) (88.45% accuracy)	✓ For five ambiguous words, only two meanings are clarified.
Solomon Makonnen (2014)	To investigate the application of machine learning techniques to word sense disambiguation of Amharic texts.	Naïve Bayes. (70-85% accuracy)	✓ Accuracy is not good enough.
Tesfa Kebede (2013)	To develop Word Sense Disambiguation for the Afaan Oromo language.	Naïve Bayes (79% accuracy)	✓ The model predicts more than two senses with a small amount of data. ✓ This reduces the generalizability of a model
Temesgen Tadesse (2021)	To design and develop WSD for the Wolaita language using a machine-learning approach.	SVC, Bagging, SVC (83.22% accuracy)	✓ Only for five ambiguous words ✓ Maximum sense of the word is three
Amlake Aschale (2021)	To find the sense for six ambiguous words in Geez language	ADTree and Bagging size 4 (92.1 Accuracy)	✓ The model only predicts two senses for ambiguous words.

2.5.3. Summary of Related Works

Most of the former researchers especially local researchers have taken five different words in their languages but as we have seen in their study the words had only two and three senses and also most of them built the model without a feature extraction method, which reduces the performance of the system. When the amount of data is too small then it reduces the generalizability of a model and in most of the studies, the model was trained with only one classical learning algorithm and didn't compare with other algorithms. Most of the local language researchers conducted their experiment for words that have no more than three senses, in our study, we tried to conduct the word with five different senses. In this study, we also evaluated the optimum window size for the Wolaita language WSD model concerning accuracy value and computational time. We also discussed the effect of stopword removal on time complexity and stemming in the performance of the models.

2.6. Overview of the Wolaita Language

The Wolaita zone is found in western Ethiopia. There are 400 kilometers between it and Addis Ababa. According to WAKASA (Motomichi, 2014), the Wolaita people call their native tongue "Wolaytta" (wolaittattuwa in their language). Wolaytta language is a language spoken by the people of the Wolaytta zone. Wolaita has been written in Latin script since the 1940s when the Sudan Interior Mission developed a system for doing so. Now, the language has advanced, and many books, dictionaries, and applications such as the Wolaita Holy Bible app and the Wolaita English Dictionary app have been developed. Additionally, the language is taught by Arbaminch Teaching College at the Diploma level and Wolaita Sodo University at the BA level (Temesgen, 2021). Wolaytta is an Afro-Asian language spoken by more than 3,000,000 people (Beldados, 2013). It is spoken by the Wolaita people as well as several other parts of Ethiopia's Southern Nations, Nationalities, and People's Region, including Gamo, Gofa, Mello, Kucha, and Dawro (Lessa, 2003). It is because the inhabitants of Wolaytta are surrounded by people who speak another Ometo dialect and hence an idiom very similar to Wolaytta (such as the Dawro, K'uc'a, Borodda, and Gamu) (Beldados, 2013).

2.6.1. Morphology of the Language

Morphology is a field of linguistics that investigates and describes the formation of words (Workineh, 2020). Similar to Amharic, Wolaita has two types of morphemes: free and bound morphemes. A bound morpheme does not occur as a word on its own, but a free morpheme can stand alone as a word. The two fundamental methods of word construction in Wolaita are affixation and compounding. Affixation is the process of adding/attaching affixes to the root, which serves as a base. Affixes are morphemes that cannot occur on their own. Affixes are classified into three types: prefixes, suffixes, and infixes. Wolaita does not have a prefix or an infix. Suffixation, on the other hand, is the primary method of word creation in Wolaita. Adding one suffix to another to construct a word is prevalent in Wolaita. Adding one suffix to another suffix can result in a somewhat long word that often contains semantic information equivalent to an entire Amharic phrase, clause, or sentence (Workineh, 2020).

2.6.1.1. Personal Pronouns

Personal pronouns are pronouns that are associated with a specific grammatical person, such as a first-person (as in I), second-person (as in you), or third-person (as he, as she). Personal pronouns can also take several forms based on their number (typically solitary or multiple), grammatical or natural gender, case, and formality. Personal pronouns are used to make basic differences in most languages. These distinctions can be found within the fundamental set of independent personal pronouns (Mara, 2018).

Table 2. 2 Personal pronouns of the language (Mara, 2018)

Wolaytta	English
tanni	i
a	she
i	he
nuuni	we
tana	me
nuussi/nuna	we
eta	them
eti	they

2.6.1.2. Morphemes, Phonetics, and Syllable Structure

Morphemes are classified into two types: free and bound. Free morphemes can stand alone as a word, whereas bound morphemes cannot. The Wolaytta is distinguished by the presence of both sorts of morphemes. Words in Wolaytta can be generated using the affixation and compounding processes. Affixes are morphemes that can stand on their own as words. Prefix, suffix, and infix are three types of affixes. Wolaytta's word construction method is solely based on suffixation. A single word in Wolaytta might have several different meanings. This is because the Wolaytta word is constructed by successive suffix additions, resulting in an unusually long word. Wolaytta syllables can be written as CV (V) (C), where C and V stand for consonant and vowel, respectively. Consonant and vowel in parenthesis indicate an optional syllable fragment. As a result, the universal syllable formulation applies to any position in the word (Beldados, 2013).

2.6.1.3. Punctuation Marks in Language

Words in the Wolaita writings are divided by white spaces in the same way that they are in English. Different punctuation signs follow the same punctuation pattern used in English and other languages that employ the Latin writing system. For example, a comma (,) is used to separate a list of ideas, concepts, names, items, and so on, while a full stop (.) in a statement, a question mark (?) in an interrogative sentence, and an exclamation mark (!) in a command or exclamatory sentence signals the end of a sentence (Temesgen, 2021).

2.6.1.4. Phonology of the Language

Phonology is a discipline of Linguistics that investigates the physical sounds of human speech. It is concerned with the physical qualities of phonemes as well as the physiological processes that produce them. Wolaytta's speech sounds, like those of other languages, fall into one of two categories: consonants or vowels (Habtamu, 2010).

Consonants: - If two sounds in a pair of words change their meaning, they are termed different phonemes. Wolaytta features 29 consonant phonemes, including voiced glottalized consonants, which have been classified as consonant clusters (Wakasa, 2008).

Table 2. 3 Wolaytta Consonantal Phoneme Inventory (Habtamu, 2010)

Voiceless	Stop: p, t, k, ʔ (?)
	Fricative: s, sh, h, nh
	Affricate: ch
Voiced	Stop: b, d, g
	Fricative: z, zh
	Affricate: j
	Nasal: m, n
	Liquid: r, l
	Approximant: w, y
Glottalized	Voiceless: P (ph), T (x), K (Q), C (c)
	Voiced: D (dh), L M, N

Vowels: - Vowels are spoken sounds produced when there is no impediment to airflow in the oral cavity. The Wolaytta language recognizes five vowels as vowel phonemes: /i/, /e/, /a/, /o/, and /u/.

Diphthongs: - Diphthongs are sounds created by combining two vowels in the same syllable. Wolaytta is composed of four diphthongs: /ai/, /au/, /oi/, and /ui/. (Wakasa, 2008). The words containing these diphthongs and their phoneme equivalents are listed below (Habtamu, 2010).

7oicc-íis	‘oi cc ii s’
kóir-o	‘k oi r o’
súíK-iis	‘s ui K ii s’
suill-íya	‘s u ill i y a ‘

2.6.1.5. Ambiguities in Wolaytta Language

Ambiguity is a characteristic of any thought, idea, statement, or claim whose meaning, intention, or interpretation cannot be resolved decisively according to a rule or process with a finite number of steps. Specific and unique interpretations are permitted in ambiguity (though some

may not be immediately obvious), whereas it is difficult to create any interpretation at the necessary level of specificity with hazy information.

Lexical ambiguity arises when a word has many meanings in a particular text, whereas syntactic ambiguity happens when the order of words has multiple grammatical relationships that modify the meaning of the text (Segid, 2015).

Semantic ambiguity: - It determines the various meanings of a sentence by focusing on the interconnections between the sentence's word-level meanings. Polysemy, idiomatic, and metaphorical sentence elements cause it. Words with numerous meanings must be semantically disambiguated. Semantic disambiguation allows only one sense of a polysemous word to be chosen and incorporated into the sentence's semantic representation (Segid, 2015).

Due to the **polysemy** that most words have, which has a variety of meanings, many ambiguous words can have several meanings by emphasizing certain letters in the word while reading. The first problem arises again whenever these terms are used in natural language-processing (NLP) systems since polysemy is a word with several meanings and, as a result, leads to the creation of ambiguity issues.

For example, “**Dummiya dawusu**”

- i. There is a special
- ii. Dumme was alive.

The ambiguity is due to / Dummiya / which refers to “special” or a female person named “Dumme”.

Idioms: are used to describe expressions that have meanings other than the literal interpretations of their constituent elements. For example, “**De”oy deexes**”

- i. Life is harsh.
- ii. Life is heavy.

The ambiguity is because of / **deexes** / which may refer to ‘harsh’ or refers to the weight as “heavy”.

Categorical Ambiguity: Occurs when lexical elements belong to various word classes yet with the same phonological and homographic form. We can explain it by using the following ambiguous words: for example, “Wolayttan **maatay** keehin bonchchettes” is interpreted as:

- i. In Wolaita the right was highly respected.
- ii. The grass was highly respected in Wolaita.

The source of ambiguity is / **maatay** / which has the verbal meaning of ‘right’ in the first sentence and a noun meaning of ‘grass’ in the second sentence.

Structural Ambiguity: The most prevalent sort of ambiguity in language is structural ambiguity, which occurs when phrases have more than one conceivable structure.

For example, ‘Ta keettaa goday anjjiis’ and ‘Ta goday keettaa anjjiis.’ can be interpreted as (Temesgen, 2021)

- i. ‘The owner of my house blessed me.’
- ii. ‘The Lord blessed my house.’

Phonological Ambiguity: ambiguity can arise from the interpretation of speech sounds within and across words. Phonological ambiguity occurs when speakers enunciate by making a pause sound. Without pauses, speaking results in word ambiguity. The following example illustrates the ambiguity:

- i. [maarin+dos]
- ii. [maarindos]

Sentence (i) is interpreted as ‘we are alive on his mercy’ with pause and without the pause as in sentence (ii), we interpret it as ‘the name of the person’.

Referential Ambiguity: is also another instance of ambiguity is when pronouns include a specific antecedent with a different meaning. For example, “Maani Bakkalo qulppiyaa bashshiis. I keehippe yiilottis”. This example can be interpreted as:

- i. Mana lost Bakkalo’s key. Therefore, mana was upset.
- ii. Mana lost Bakkalo’s key. Therefore, bakkalo was upset.

The above examples are interpreted as: ‘I’ refers to both ‘mana’ and ‘bakkalo’ in the sentence.

CHAPTER THREE

2. RESEARCH METHODOLOGY

This section describes the steps taken to accomplish the goal of the suggested study. and it mainly discusses the study's design, the techniques we'll employ to carry it out, and the methodology we used. This section also covers data sources, data annotation, preprocessing techniques, tools (hardware, software, and deployment), feature sets that will be utilized for model implementation, and performance metrics that will be used to assess the Wolaita language WSD model. To achieve the general and specific objective of our study, we used a quantitative experimental research design to select the best and outperforming models and feature extraction techniques.

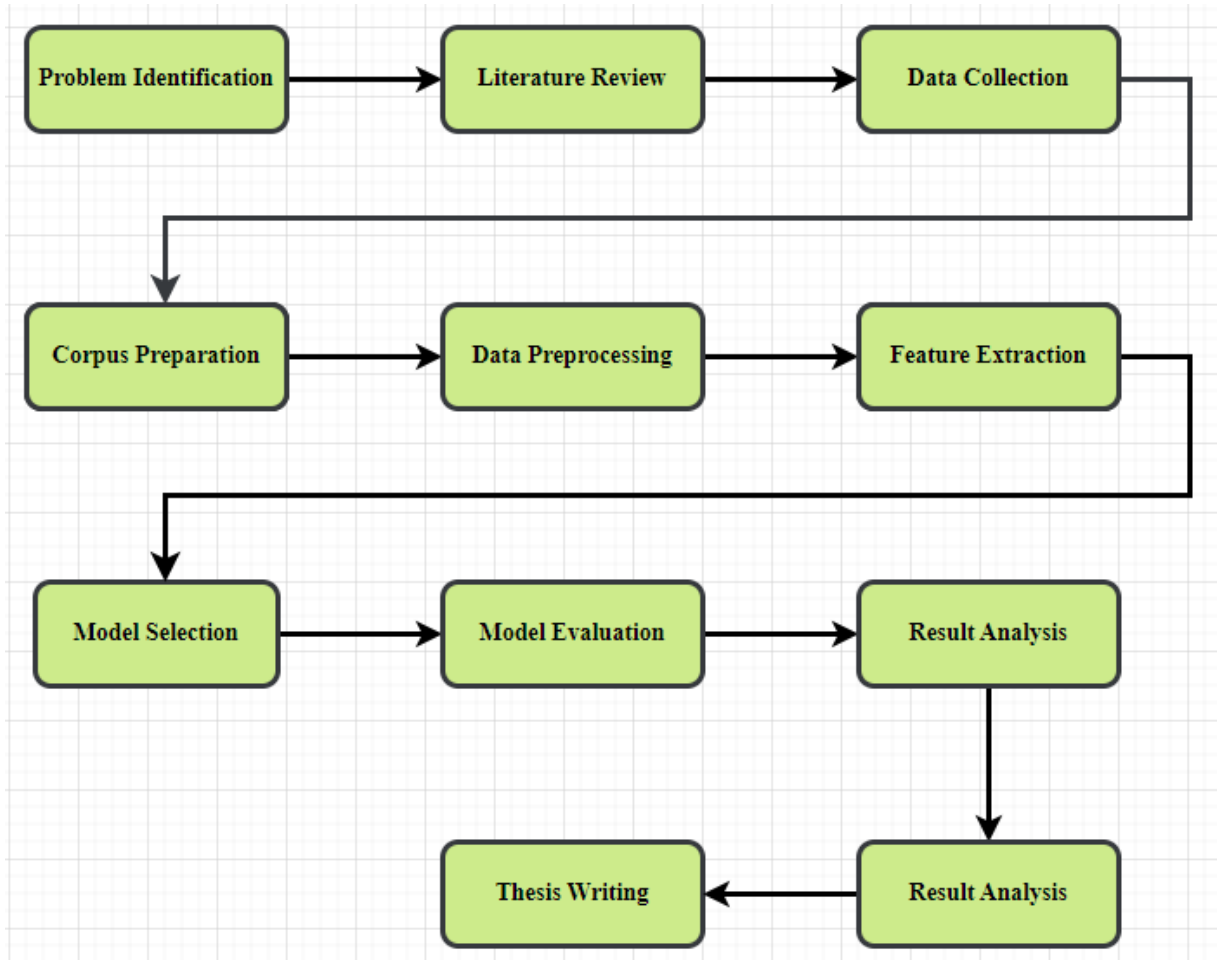


Figure 3. 1 Steps of research methodology

3.1. Data Collection

As we have used a corpus-based approach in this study, due to the absence of a common-sense annotated corpus or Wordnets for the Wolaita language, obtaining sense annotated corpus for NLP applications like WSD is challenging. Therefore, data collection takes precedence over data preparation. Thus, we collected the data from different sources like media (Wolaita Fana FM 99.9, Wogeta FM 96.6, and Wolaita TV), social media (Wolaita Times from Telegram channel), Online Holy Bible, Wolaita Language and Literature department, academic texts, and previous researchers. Data collected from multiple sources can help to eliminate the imbalance between meanings (senses) of polysemous words. As Temesgen (Temesgen, 2021) discussed, the data annotation was done by the Wolaita language lectures in WLLD at WSU. Annotators are chosen based on their ability and academic level to annotate data. We also collected and prepared the data for the new two words based on how they annotated the data in prior research and how they annotated based on their criteria.

3.1.1 The Main Sources of Data

Academical sources: We gathered data from the different education-level Wolaita language textbooks. In addition to textbooks, we collected the data from modules that are given by the Wolaita language and literature department at Wolaita Sodo University.

Online sources¹: We collected data from different online sources which are stored on websites by previous Wolaita language researchers. These data were acquired from free and open-source machine translation platforms offered by researchers.

From Holy Bible²: We gathered data from the online Holy Bible which is written in the Wolaita language and also from the Wolaita language Holy Bible (named wbible).

From Media: We also collected data from different mass media and social media. Mass media like Wolaita Fana FM, Wogeta FM, and Wolaita Tv, and social media like Wolaita Time from its Facebook and Telegram channel in Amharic and translated to the Wolaita language.

As we discussed in the previous chapter, the supervised learning-based WSD was done for the Wolaita language by Temesgen (Temesgen, 2021) in which he prepared the corpus from five

¹ <https://sourceforge.net/p/apertium/svn/77956/tree/incubator/apertium-wal/texts/>

² https://ebible.org/study/?w1=bible&t1=local%3Awal&v1=JN1_1

ambiguous Wolaita words. In addition to the words used by the researcher, we added two words to add the corpus size. (Temesgen, 2021) built 2797 sentences from scratch, with each ambiguous word having 100 or more sense sentences. Similarly, we collected 1231 sentences for two added ambiguous words and we used 1060 sentences in addition to the 2500 sentences adapted from Temesgen, for a total of 3560 sentences for seven ambiguous words, each of which has three and more than three meanings. We acquired 2500 sentences from the previous researcher and made them suitable for our study as the data we adopted is not well-organized.

In general, we collected the Wolaita words from several sources for our study. We collected seven ambiguous words as data requirements (see table 3.1), which are balanced and distributed, which means that each sense has roughly the same size of sentences because the accuracy of machine learning algorithms degrades significantly when the distributions of the senses in the training and testing samples differ. As a result, we prepared 3560 sentences for seven ambiguous words. The total number of senses for seven selected ambiguous words was 24, this makes our study the maximum numbered sense study among other local language WSD studies.

Table 3. 1 Statistics for datasets

Words	Sample sentences	Number of meanings	Number of meaning examples of each sense				
			Sense 1	Sense 2	Sense 3	Sense 4	Sense 5
Doonaa	500	3	172	164	164	-	-
Naaga	500	4	126	122	130	132	-
Hayttaa	500	3	175	140	165		
Ayfiyaa	500	3	150	180	170		
Ogiyaa	500	3	164	172	164		
Sinttaa	560	5	114	110	113	111	112
Aahdha	500	3	174	167	159		
Total	3560	24					

3.1.2. Inter-annotation using Fliess kappa

The Fleiss' kappa statistic is used to determine the agreement among several annotators or raters when categorizing or labeling materials into multiple categories. It is widely utilized in studies of inter-annotator agreement in a variety of domains, including natural language processing

(NLP). In this study, we selected five annotators and we evaluated the measure of reliability by using SPSS software to check whether they applied the data annotation guideline or not. We calculated kappa (k) values and we obtained $k = 0.84$ as shown in Appendix IV. This kappa value implies the agreement between the raters is almost perfect, as the guideline if the $k=0.81-1$, then the interpretation is almost perfect agreement.

3.2. Corpus Preparation

A corpus is a group of texts used for linguistic analysis, typically kept in an electronic database to make the information conveniently accessible through a computer (Miller et al., 1993). This information represents the lexical sample task data. It is typically a huge number of naturally occurring phrases containing a specific target word, each tagged with a pointer to a sense item from the sense inventory. Authentic (naturally occurring) words, whether spoken or written, make up the majority of corpora (Feyisa, 2015). Corpus will be expected to have the following characteristics, according to (McEnery & Wilson, 2001) are:

Machine-readable: Although it is not always the case, machine-readable corpora are now believed in many contexts.

Sampling and representativeness: due to the enormous number of words in many natural languages, it is challenging to create a corpus that contains every word in the language. It is done by using sample words that may serve as an example of the other terms. The sample must accurately reflect the variety of words and the structural and morphological variations among them. A corpus can exist in two different forms: unannotated and annotated text. An annotated text is a collection of texts that contain grammatical or linguistic information. While an unannotated corpus is a collection of texts without linguistic information (Miller et al., 1993). Annotated texts can be used for different purposes. In linguistics, a suitably annotated (tagged) corpus can be used to study linguistic features such as the morphology and phonology of a language. It can also be used as part of voice commands and WSD (Feyisa, 2015). The corpus is provided to the system as training data, so the system can learn/adapt some patterns from the corpus for each word or sentence. The corpus size affects the learning propensity of the system. The larger the corpus size, the greater the learning bias of the system.

In general, this large size of corpus preparation is challenging and time-consuming work. Finally, we created the Wolaita language corpus manually and 2500 from (Temesgen, 2021).

First, we had to select and collect contextual ambiguous words with three or more meanings from the Adiel Wolaita dictionary. When determining which words to use in WSD, the most common words, and their meanings were carefully selected. A word's meaning is a generally accepted meaning of a word. For example, let us see the following examples:

✚ Bassi ba oosuwaa **ayfiyaa** naagiidi de'es.

✚ Na'ay ba ishshawu **ayfiyaa** baqqiis.

The word “**ayfiyaa**” is used in the above sentences with two different meanings/senses:

It is used in the first sentence as **murutaa (result)** and in the second sentence, it is used as **bollaa kifiliyaa (eye)**. Defining a word's store of meaning is a key problem in distinguishing word meanings. A meaning inventory divides a word's meaning range into its meanings. Then, the ambiguous words are searched from different textually written Wolaita language documents written.

3.3. Data Preprocessing

Data preprocessing refers to the actions required to change or encode data so that it may be easily interpreted by a machine. The algorithm should be able to easily interpret the data's attributes for a model to be accurate and exact in predictions. In this preprocessing, we may follow the techniques to make the data into readable form for the algorithms such as tokenization, stop word removal, data cleaning, and normalization. We addressed missing values in our study by replacing them with zero. Because of the morphology of the language, some words from the left and right of target words may be overlooked in our datasets depending on window sizes. As a result, these missing values were replaced with zero using Python.

3.3.1 Data Cleaning

It is one of the data preprocessing techniques which is used to remove unessential things that can affect the model performance and mainly it removes the special character that is in the data collected like symbols, punctuations, symbols, and other irrelevant characters.

1. *Read corpus (file)*
2. *Define word delimiter to space*
3. *Read each sentence from corpus (file)*
 - While file delimiter*
 - If there is word delimiter*
 - Open new file*
 - Put each term as separate tokens in new file*
 - Endif*
4. *Close file*

Figure 3. 2 Tokenization algorithms (Feyisa, 2015)

3.3.3 Stop Word Removal

Not all terms in the document (or corpus) are equally crucial for disambiguation. A few words appear often in most documents. As a result, it's crucial to get rid of any concepts that aren't relevant to figuring out what an ambiguous word means. There are two ways to get rid of these terms. The first technique involves counting the number of occurrences of high-frequency phrases to delete them. (frequency). Utilizing the language's stop word list is the second technique. Therefore, stop word removal was applied in this investigation using the second approach. Prepositions, conjunctions, and articles are examples of the Wolaita language domain-independent stop words. These terms must be eliminated during the preprocessing stage. As a result, we utilized stop words for the Wolaita language that were gathered by earlier researchers (Lessa, 2003). Reducing the input text's quantity, and eliminating stop words in WSD may improve time complexity. Stop words only make up a small portion of the whole text, hence there is typically little impact on space complexity. The precise effect will rely on the WSD algorithm's implementation specifics and the input text's qualities.

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 the stop word list Then Remove a word

 End if

End for

3. Return the remaining words

4. HALT

Pseudo code 3. 2 Pseudo code for Stop word removal (Ermias, 2021; Temesgen, 2021)

3.3.4 Normalization

The technique of normalizing alphabets that share the same sound and meaning but use various representations of the alphabet is known as normalization. The process of transforming a token into its original form is known as normalization³. A word's inflectional form is eliminated during the normalizing procedure to produce the base form. Normalization is useful for removing variations from a text and lowering the number of unique tokens that are there. Also editing the wording by eliminating unnecessary details. Normalization approaches assist minimize the dimensionality of text data, manage lexical variances, and assure consistency among documents. These preprocessing stages create the groundwork for more accurate and successful NLP tasks like text classification, information retrieval, sentiment analysis, word sense disambiguation, and machine translation.

³ <https://www.analyticsvidhya.com/blog/2021/03/tokenization-and-text-normalization/#:~:text=Normalization%20is%20the%20process%20of,form%20of%20antinationalist%20is%20national.>

WSD Normalization Algorithm

Input: WSD dataset

Output: normalized WSD dataset

INIT:

1. Read the WSD dataset

2. WHILE (it is not the end of the file):

IF the 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

Pseudo code 3. 3 Pseudo code normalization (Ermias, 2021; Temesgen, 2021)

3.3.5 Stemming

The process of reducing morphological variants of words into the base or root forms is known as stemming. A stemmer will significantly improve WSD systems in morphologically rich languages like the Wolaita language. Because of their grammatical function, different concepts in the Wolaita language are formed from the same root word. To generate various derivational and inflectional word forms Prefixes, suffixes, and infixes are used in the Wolaita language. As a result, any extra words or characters that modify the root word to new forms are stemmed from the corpus using a stemming method designed by (Amlakie, 2020), that is appropriate for the language. This algorithm just eliminates prefixes and suffixes because we created stemmer affix removal. The algorithm mostly removes suffixes from our datasets as the Wolaita language is rich in words with suffixes rather than prefixes. As a result, to obtain the common form of the ambiguous word, we attempted to manually normalize root word infixes (Amlakie, 2020).

Derivational suffixes are typically used to generate new words in a language from the stem or base form of an existing word. Words can alter in tense, case, a plurality (number), and gender depending on their inflectional suffixes (Feyisa, 2015). For example, the ambiguous word “Sinttaa” may take as, ‘sinttappe’, ‘sinttawu’, ‘sinttan’, ‘sinttassi’, etc. after the removal of the suffixes, and to make the ambiguous words readable (suitable) for machine learning algorithms, we manually look up all these words for words that is “Sintta”. The same approach was taken concerning other ambiguous words that were used in this research, but not concerning context words. Due to the time required to normalize all of the context words utilized in this study, normalization was not applied following stemming for the context terms.

```

Read corpus file from directory
Read prefix list
Read suffix list
Assign the first 1, 2, 3,... character(s) of the token to prefix
Assign the last 1, 2, 3,... character(s) of the token to suffix
For each token in file
    If token prefix is in prefix list then
        Remove prefix from token
    End if
    If token suffix is in suffix list then
        Remove suffix from token
    End if
Endfor

```

Pseudo code 3. 4 Pseudo code for stemming (Amlakie, 2020)

3.4. Feature Extraction

Feature extraction in textual corpora is the process of turning a dataset’s list of words into a feature vector that machine learning models find appealing. As a result, feature vector representations of datasets that had successfully undergone preprocessing were created for use by algorithms. We created a list of features by taking into account the morphological structure of the Wolaita language that could be useful in resolving the word sense disambiguation problem. The models selected cannot be directly trained on the corpus with the chosen ambiguous words. By considering the above reasons we have chosen four feature extraction techniques to convert text into vectors, the techniques are BOW, Word2vec, and Tf-Idf integrating with N-gram.

3.5. Model Selection for the Wolaita Language WSD

The loaded dataset was preprocessed and represented as feature vectors before being input into the selected classification machine learning models for our study. We have selected six different classification algorithms for this study from the supervised machine learning approach which are: Neural Network, AdaBoost, Bagging, SMV, Random Forest, and NB. In the case of unsupervised learning, we have selected four different clustering algorithms which are: EM, Hierarchical clustering, simple k-means, and Farthest First. We selected those algorithms because they had the best performance in different local languages such as Amharic, Afaan-Oromo, Tigrigna, and Geez language in both of the approaches. We selected multi-class supervised classifiers to classify our datasets namely Bayes classifier (NB), MLP classifier (NN), RF classifier (RF), SV classifier (SVM), Bag classifier (Bagging), and Adb classifier (Adaboost). Additionally, the majority of the chosen algorithms are ensemble techniques that can deal with difficulties in the datasets including overfitting and data instability. To increase the performance of the selected models we used different techniques discussed below:

Weak Learners (Base Estimators): as most of the models selected are ensemble, these methods use base estimators and add the weights of each weak learner to improve the performance. For example, the bagging classifier uses a decision tree (DT) as the default weak learner. In addition to DT, we use another estimator which is Linear SVC.

Optimization techniques: to improve the performance of the model, we also optimize the process, which improves the accuracy of machine learning by decreasing the error degree. There are two basic known techniques for optimization i.e., Gradient Descent (GD) and Stochastic Gradient Descent (SGD). Among these two techniques, we choose SGD for this study as the time to calculate the optimum vector becomes very large in GD. To overcome the limitation of GD, SGD techniques were proposed. In SGD, the local minimum of the function is computed using a sample rather than the entire set of data points, and as a result, it is faster than gradient descent. The most known challenge of optimization is overfitting. To overcome this challenge, we used the technique called regularization in this study. It has two basic techniques namely L1 and L2 regularization. In this study, we used L2 regularization because It alters the loss function by including a penalty term that is proportional to the square of the weights. By decreasing them toward zero, this penalty term pushes the model to employ smaller weights.

The model's generalization capabilities can be enhanced and its complexity can be reduced as a result. Instead, for correlated words, it is important to assign small weights to features through L2 regularization as (Temesgen, 2021; Ying, 2019) claimed in their study.

3.6. Model Evaluation Metrics

In this work, two methods were used to evaluate and test the model. These are the k-fold CV and performance metrics such as the confusion matrix, accuracy, precision, recall, and F1-score. The reasons for choosing k-fold CV are: is used as a major technique by exalting the model's ability to predict unseen data increases by k times the CV, mitigating the problem of CV overfitting. This is also a popular scoring method. That's why we chose the k-fold CV. We used 10-fold CVs from k-fold CVs, 9 folds were used for training, and 1-fold was used for iterative testing. Bias in the model decreases as the number of k values increases (Marcot & Hanea, 2021; Temesgen, 2021).

CHAPTER FOUR

4. PROPOSED WOLAITA LANGUAGE WSD MODEL

4.1. Overview of the System

This session describes the overall design and implementation of the proposed model. This chapter also describes the general architecture of the proposed system, the design proposed preprocessing techniques we used in the study, the design of feature extraction for the proposed model, and the implementation of the proposed models.

4.1.1. Development Tools

To design and implement the suggested thesis work, a variety of development tools will be used in this study. These tools include the research's overall relevant tools. The sections that follow provide a quick overview of these development tools.

4.1.2. Design Tools

The development, interpretation, and presentation of design concepts will all be done using design tools. The suggested system will be created using draw.io. It is a portable and effective visual design tool for making flowcharts, network diagrams, flowchart diagrams, and other diagrams that look good.

Hardware tools

The following are hardware tools which are used in the implementation.

Table 4. 1 Hardware tools

No	Tools	Importance
1	GPU	to enhance computation and speed up training
2	Hard Disk	used to store huge datasets.
3	RAM	to accelerate the training process cooperatively with GPU

Software tools

For implementing the research by coding, writing different software and coding tools will be used and those are listed below with their description.

Table 4. 2 Software tools

Tools	Descriptions
Anaconda navigator	it is a graphical user interface for the Python desktop that offers conda packages, environments, and platforms.
Jupyter Notebook	it is a web application written in Python that generates live code, visuals, and narrative prose.
Spyder (Anaconda)	the Spyder (Anaconda) is also used to write Python code.
Python	a simple but powerful programming language for creating a machine-learning application.
MathType	used to write different formulas and symbols
Microsoft Excel	used for annotation and data preparation operations such as cleaning, filtering, and sorting
Microsoft Word	used to write the documents
SPSS	to evaluate the Kappa value

The following Python packages are going to be used in the implementation of the proposed system. These packages were imported by Python and performed a specific function.

Table 4. 3 Python packages

Packages	Description
NLTK	The platform is used to handle data involving human language.
NumPy	Used in Python to parse multi-dimensional arrays
Flask	It is a microweb framework that is used to create websites.
Pandas	Utilized to alter, read, and write data frames
Scikit-learn	Integrated Python module for classical machine learning techniques
Pickle	Python objects are used to be stored and converted in a database or file.
Matplotlib	Used in Python to display data
Seaborn	Used in Python to create visually appealing data visualizations.

4.2. The Architecture of the Proposed System

The general architecture of the Wolaita language WSD contains different steps as in the figure below. At first, it contains the overall raw data collected from different sources, from the raw data we selected sentences that contain the selected ambiguous words. The second one is it contains the different preprocessing techniques such as the tokenization process, stopwords removal, stemming, and normalization techniques. Then the preprocessed data are converted into the vector form. For conversion purposes, we have used different feature extraction techniques such as BOW, Word2vec, and TF-IDF with N-gram. Now we have the annotated corpus. After the completion of the conversion the proposed machine learning models (Naïve Bayes, SVM, Bagging, Neural Network, Random Forest, and Adaboost) for the supervised approach and (EM, simple KM, HierarchicalClusterer, and Farthest First) for the unsupervised approach were trained. The models are evaluated by using different evaluation techniques. Finally, the given sentences were disambiguated.

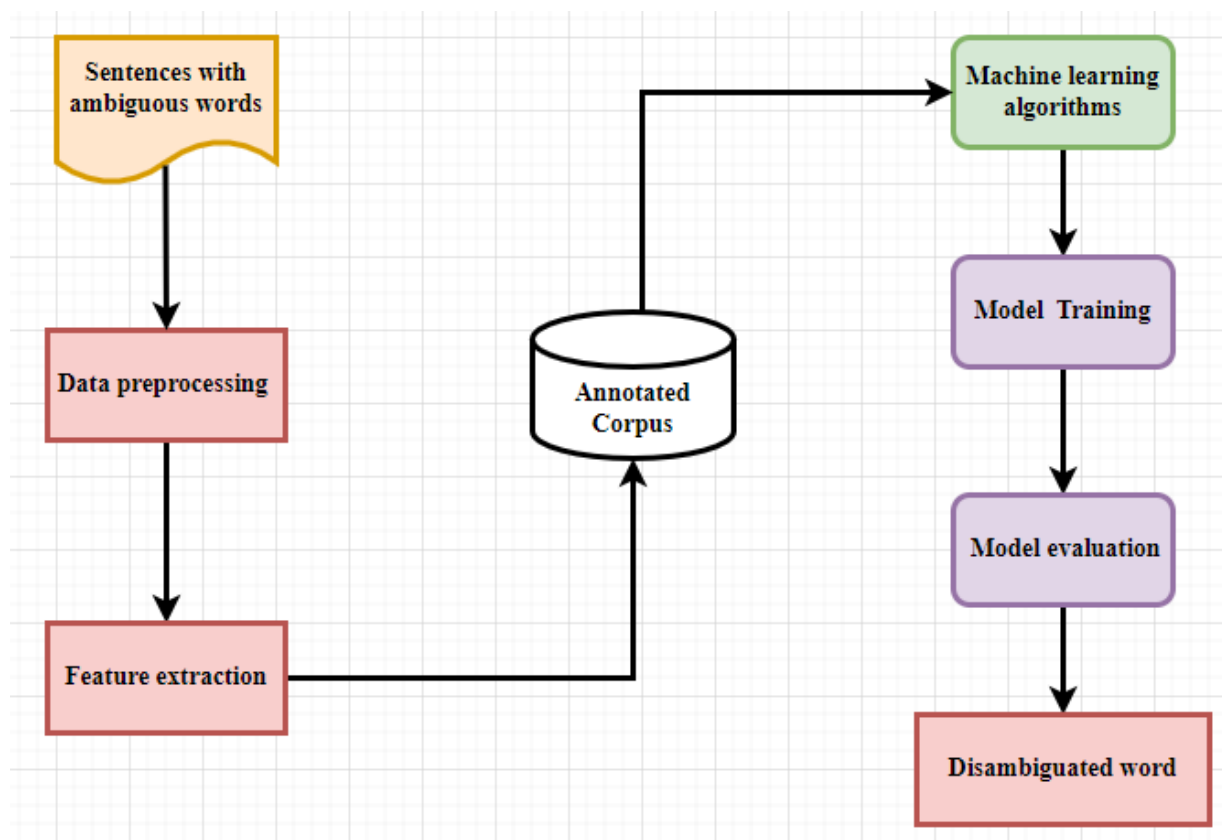


Figure 4. 1 Proposed Supervised Approach Architecture of Wolaita Language WSD

4.3. Proposed Preprocessing Techniques

Data is the primary resource that we must collect from many sources and assemble as a corpus for various studies, particularly in the linguistic sector. So, to make the data readable for the machine we have to apply preprocessing techniques in NLP. For this study, the missing data are handled by replacing zero. As discussed in section 3.3 we have used four different techniques for our data preprocessing purpose (tokenization, stopword removal, stemming, and normalization) and finally, we achieve the cleaned data.

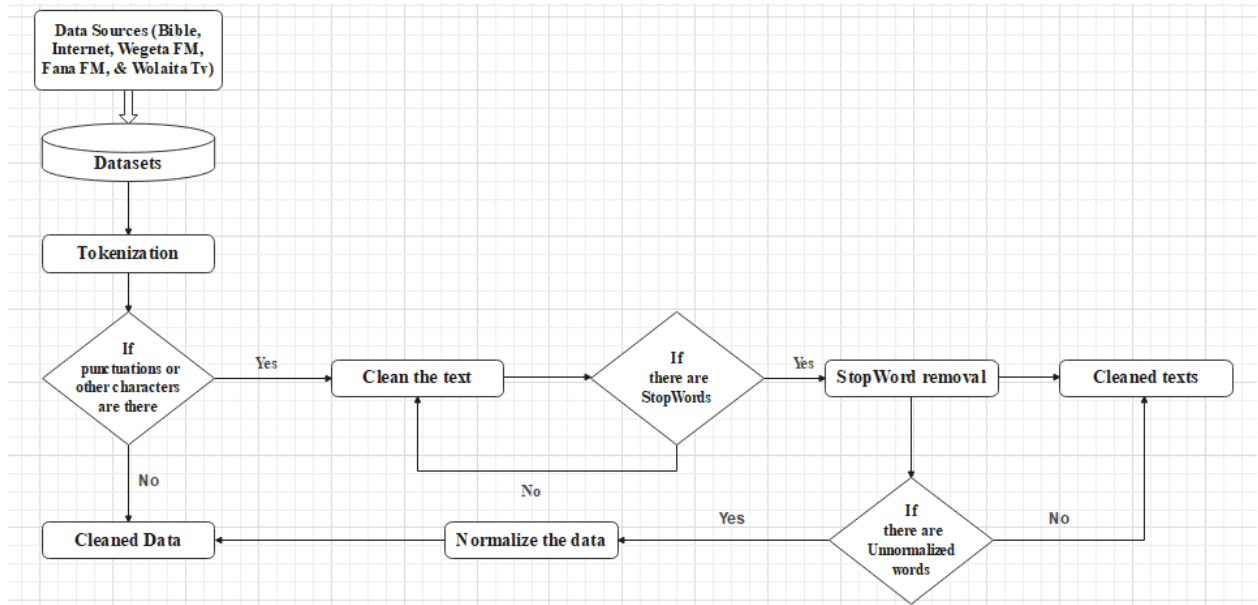


Figure 4. 2 Proposed Preprocessing phase

4.4. Proposed Feature Extraction

As after the preprocessing is done, we have to convert the preprocessed raw data into a readable form machine. The process of such a conversion technique is feature extraction. There are different techniques for feature extraction in NLP among those extraction we have chosen for techniques, named BOW, Word2vec, and TF-IDF with N-gram. In general, BOW is a straightforward method to word2vec with several limitations. The primary drawback is that the words in the given dataset are not accurately sorted as they are collected, thus much of the context from the text is lost. To circumvent this constraint, employing n-grams is the best option; it addresses concerns by providing some order to retain context.

BOW (Bag-of-Words): this method depicts text as a collection of distinct words, with no consideration for grammar or word order. It generates a vocabulary of terms from the entire corpus and displays each text as a vector, with each dimension representing a vocabulary word and the value representing the frequency or presence of that word in the document.

TF (Term Frequency): this shows how significant a word is in a document about the overall corpus. It combines inverse document frequency, which assesses the rarity or prevalence of a word across the corpus, with term frequency, which counts how frequently a word appears in a text. It has two equations: -

$$TF = \frac{\text{count of } t \text{ in document}}{\text{number of words in doc}} \quad (1)$$

The words are vectorized on the vocabulary of all possible words after the computation of TF in Equation (1). However, extraneous words are more frequent in TF than other words. IDF is used to resolve these issues.

IDF (Inverse Document Frequency): it evaluates the document's term t's level of informational depth. Stopwords in the texts, it has a low measurement. Some terms may not be in the vocabulary during query time, resulting in $df = 0$. Since integers cannot be divided by zero, Equation (2) below shows what happens when we add 1 to the denominator.

$$idf(t) = \frac{N}{df} \quad (2)$$

Equation (2) calculates the document N over the occurrence of term it, while Document N calculates $df = \text{occurrence of term } t$. This demonstrates that the frequency and useless word measures have the lowest value. When the value of $df = 0$ we will use the logarithmic equation to handle the above equation.

$$idf(t) = \log \frac{N}{(df+1)} \quad (3)$$

from the above three equations, Tf-idf is the product of Equation (2) and Equation (3).

$$tf - idf(t, d) = tf(t, d) * \log \frac{N}{(df+1)} \quad (4)$$

Word2Vec: Natural language processing (NLP) practitioners frequently extract features using the prominent Word2Vec approach for word representation. It is a method based on neural

networks that learn continuous vector representations, commonly referred to as word embeddings, from sizable text corpora. Its main goal is to represent words as dense vectors in a continuous vector space to capture semantic and syntactic similarities between them. Both the skip-gram and continuous bag-of-words (CBOW) architectures are used to learn these word embeddings in an unsupervised way.

N-grams: are ongoing groups of letters, numbers, or other symbols that appear in a document. Technically speaking, they are the adjacent groups of items in a document. They are relevant when performing NLP (Natural Language Processing) activities on text data. A Wolaita language corpus was given as input for the n-gram joint probabilities training stage. By taking into account these joint probabilities, it may be assumed that target ambiguous words will be given the right sense during the testing stage. Finding the sense of supplied ambiguous words that has the highest joint probabilities with their co-occurring words gathered using uni-gram, bi-gram, and tri-gram techniques in the training corpus is the objective of n-gram-based WSD.

TF-IDF with N-grams: TF-IDF and N-gram are two strategies for feature extraction in NLP tasks that can be employed separately. They can, however, be integrated and utilized together for certain purposes, such as WSD, to improve the representation of textual data. Integrating TF-IDF with N-grams in the context of WSD can help capture both the relevance of terms and the local context in which they emerge. N-grams allow you to collect word sequences that may provide important context for disambiguating word meanings. The TF-IDF algorithm can then be used to weight the N-grams based on their relevance in the document or corpus. Integrating TF-IDF and N-gram for feature extraction in Word Sense Disambiguation (WSD) can help capture both local and global context information, enhancing WSD task accuracy.

4.5. Proposed Classifiers for the WSD Model

As our study is the comparison of machine learning algorithms for word sense disambiguation, firstly we applied four classifiers to our datasets. The classifiers are important for the good performance of the models by classifying the data as readable and convenient for the selected machine learning algorithms. The six built classifiers are Bayes classifier, SVC, Bagging, AdaBoost, MLP, and Random Forest classifiers for the model building, from six different algorithms, namely Naïve Bayes, SVM, Bagging, AdaBoost, Neural network, and

Random Forest for the labeled data. In addition to this, we tried to conduct the algorithms like simple k-means, FarthestFirst, Expectation matrix, and Hierarchical Cluster.

Bayes classifier: is a classification method that uses Bayes theorem to find the most likely meaning of a word in a particular situation. The goal of WSD is to discern the correct sense or interpretation of an ambiguous word depending on its context. To classify an ambiguous word using the Bayes classifier, the algorithm computes the posterior probability for each potential sense of the word and chooses the sense with the highest probability as the forecast sense. The prior probability can be estimated using the frequency distribution of the senses in a training corpus, whereas the likelihood of detecting the context given a sense can be evaluated using multiple strategies such as counting co-occurrences or machine learning algorithms.

Multilayer Perceptron (MLP) classifier: this is a kind of artificial neural network that can be used for Word Sense Disambiguation (WSD). It is a popular choice for numerous classification tasks, including WSD, given its ability to learn complicated patterns and correlations in data. In the context of WSD, the MLP classifier can be trained to predict the proper sense of an ambiguous word given its surrounding context.

Bagging classifier: this also known as Bootstrap Aggregating, is a machine learning ensemble method that can be used to solve NLP and other problems. Ensemble approaches integrate the predictions of numerous independent classifiers to get a final prediction, which often improves accuracy and robustness. It can help WSD classifiers perform better by overfitting reduction, boosting stability, and capturing a broader range of information in the training data.

Adaptive Boosting (AdaBoost) classifier: is another ensemble method that can be used for Word Sense Disambiguation (WSD) tasks. Adaboost is well-known for its ability to combine poor classifiers to produce a strong ensemble classifier. This classifier is especially useful when weak classifiers outperform random guessing but still have low individual prediction power. Adaboost can generate a strong ensemble classifier that performs better in WSD tasks by iteratively changing the instance weights and combining the weak classifiers.

Random Forest classifier: is a well-known ensemble approach for Word Sense Disambiguation (WSD) applications. It is a bagging approach extension that employs decision trees as foundation classifiers. Random Forests are useful in a wide range of classification issues,

including WSD. It provides various advantages for WSD, including the ability to handle huge feature spaces, reduce overfitting, and capture complicated patterns in data. The Random Forest ensemble of decision trees may create accurate meaningful predictions by utilizing the strengths of individual decision trees.

Support Vector Classifier: this is a machine-learning technique that can be used for different problems. SVC is a form of Support Vector Machine (SVM) technique that is often used for binary classification tasks. However, it may be extended to handle multi-class classification tasks, which is useful for WSD because several senses must be predicted. SVC is well-known for handling high-dimensional feature spaces and capturing complex decision boundaries. It is capable of handling multi-class classification jobs in WSD and has been successfully employed in a variety of natural language processing applications.

4.6. Proposed Evaluation Techniques

We utilized the major evaluation technique for the clustering algorithm in this study. The ability to generalize the clustering results would be evaluated using this accuracy. We must set a limit on the number of clusters when we use the chosen clustering algorithms for our datasets. To balance the number of ambiguous word senses, one method is to limit the number of clusters. That makes it easier to identify which cluster each instance belongs to. Using the clustering assumption again helps in preparing a fully labeled dataset for classification.

To evaluate the classification and boosting algorithm's training and test of the model there are two techniques are applicable which are the performance evaluation techniques and K-fold CV. There are various methods for assessing performance. We used the most popular evaluation techniques to evaluate how quickly our algorithm resolved ambiguities. These measures include accuracy (A), recall, F1-score, and precision (P) as the K-fold CV was used by the former researcher of the Wolaita language.

Accuracy (A): - accuracy in WSD is the proportion of instances that were properly classified.

$$Accuracy = \frac{TF+TN}{TP+TN+FP+FN} \quad (5)$$

Precision (P): - is one of the best-performing measurement techniques of the models in both traditional information retrieval and the WSD system and measures the performance of the system that provides the information with the relatedness of the information retrieved

$$\textbf{Precision} = \frac{TF+TN}{TP+FP} \quad (6)$$

F-score: - as determined by the formula below, is the harmonic mean value of both the precision and recall findings.

$$\textbf{F}_{1-score} = 2 * \frac{1}{\frac{1}{Precision} + \frac{1}{recall}} \quad (7)$$

Recall: - is an additional WSD metric that assesses how many objects are successfully categorized as well as the performance of numerous systems. In our situations, WSD is the proportion of the test set's total words that have been correctly labeled.

Confusion matrix: a square matrix in which each row corresponds to an instance in a predicted class and each column corresponds to an instance in a real-world (ground truth) class. According to the predictions of the model, the matrix's elements display the number or percentage of cases that fall into various categories. It gives the value of True positive (TP), False positive (FP), True negative (TN), and False negative (FN). True positive and True negative refer to actual positive and negative incidents, respectively, that have been accurately labeled. Negative occurrences that are mistakenly labeled are referred to as false positives while false positives are negative instances that are incorrectly labeled.

4.7. Implementation

In this section we discussed the implementation of the overall WSD for the Wolaita language, starting from importing Python packages, preprocessing phases, feature extraction, data-splitting, model evaluation, and the evaluation metrics implementation.

4.7.1 Importing Python Packages and Libraries

As we mentioned in previous chapters, we used different Python packages and libraries to implement this study. We have used a Jupyter Notebook platform to compile the code of Python and we installed different important libraries including NLTK which are essential to our study

```
In [1]: #Loading Library
import pickle
import numpy as np
import pandas as pd
import seaborn as sns
import string
import matplotlib.pyplot as plt
import nltk
from nltk.corpus import stopwords
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.metrics import accuracy_score
from sklearn.ensemble import VotingClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import precision_score, recall_score, f1_score, accuracy_score
get_ipython().run_line_magic('matplotlib', 'inline')
```

Figure 4. 3 Python Packages and Libraries for our study

```
df = pd.read_csv(r'sintaa.csv', header = None, encoding='latin-1',)
df.rename(columns={0:'LeftContext_5',1:'LeftContext_4',2:'LeftContext_3',3:'LeftContext_2',4:'LeftContext_1',5:'Target_word',
df=df.fillna(0)
df.head(10)
```

Figure 4. 4 Loading dataset to Python

```
# putting data using windowing
df['sense_example1'] = df['LeftContext_5']+" "+df['LeftContext_4']+" "+df['LeftContext_3']+" "+df['LeftContext_2']+" "+df['LeftContext_1']
df['sense_example2'] = df['LeftContext_5']+" "+df['LeftContext_4']+" "+df['LeftContext_3']+" "+df['LeftContext_2']+" "+df['LeftContext_1']
df['sense_example3'] = df['LeftContext_4']+" "+df['LeftContext_3']+" "+df['LeftContext_2']+" "+df['LeftContext_1']+" "+df['Target_word']
df['sense_example4'] = df['LeftContext_3']+" "+df['LeftContext_2']+" "+df['LeftContext_1']+" "+df['Target_word']+" "+df['RightContext_1']
df['sense_example5'] = df['LeftContext_2']+" "+df['LeftContext_1']+" "+df['Target_word']+" "+df['RightContext_1']
df['sense_example6'] = df['LeftContext_1']+" "+df['Target_word']+" "+df['RightContext_1']

df['sense_example1']
```

Figure 4. 5 Putting data using windowing

4.7.2. Preprocessing

Tokenization: is one of the building blocks of data preprocessing in NLP. It involves breaking down words into smaller units called tokens. It also makes it possible for machines to interpret and process human languages. It is a necessary step before carrying out various NLP tasks like Word sense disambiguation, text categorization, information retrieval, machine translation, and sentiment analysis.

```
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)
df['sense_example6'] = df.apply(lambda row: nltk.word_tokenize(row['sense_example6']), axis=1)
df['sense_example1']
```

Figure 4. 6 Tokenization implementation

Stopwords: are words that are frequently employed in natural languages. However, they are not required to train the model in NLP. Because training the model requires key features. We imported NLTK packages as mentioned above to get rid of Wolaita the stopword.

```
from nltk.corpus import stopwords
from nltk import word_tokenize
STOPWORDS = stopwords.words('wolaita')
def clean_text():
    #convert all data into lower letter
    df["sentence"] = df["sentence"].apply(lambda x: str(x).lower())
    #remove punctuation
    df["sentence"] = df["sentence"].apply(lambda wolaita: wolaita.translate(str.maketrans('', '', string.punctuation)))
    #remove number number
    df["sentence"] = df["sentence"].apply(lambda wolaita: wolaita.translate(str.maketrans('', '', "0123456789\!'\"'")) )
    #remove stop words
    df["sentence"] = df["sentence"].str.split(' ').apply(lambda wolaita: ' '.join(k for k in wolaita if k not in STOPWORDS))
    #tokenize sentence into words
    sentence = df['sentence'].tolist()
    map(word_tokenize, sentence)
```

Figure 4. 7 Stopword removal implementation

Normalization: as the data are collected from different sources, we have to normalize the dataset after cleaning the data by removing emojis, numbers, characters, and other unwanted punctuations. Wolaita language is also written in Latin as the English language, there are different words used in the Wolaita language as it is in English language. So, we applied normalization techniques to overcome such things.

```
In [10]: # Dictionary of words to be norimalized
word_dic = { "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))
df["sense_example6"] = df["sense_example6"].apply(lambda text: normalize(text))
```

Figure 4. 8 Normalization implementation

4.7.3 Implementation of Feature Extraction

As we discussed in the previous chapters, we used four different feature extraction techniques which are BOW, Word2vec, and TF-IDF with N-gram. The below figure shows the sample code for implementing Tf-idf with n-gram.

```
# train model with different n-gram ranges
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import f1_score
for N in range(1,11):
    # convert training data to bag of words
    cv = CountVectorizer(analyzer = 'word',ngram_range=(1,N), stop_words='english',
                        'tfidf', TfidfTransformer())
    X_train_cv = cv.fit_transform(X_train)
    X_test_cv = cv.transform(X_test)
    # train model and generate predictions
    clf = MultinomialNB()
    clf.fit(X_train_cv, y_train)
    y_pred = clf.predict(X_test_cv)
    # compute f-1 score
    score = np.round(f1_score(y_test, y_pred, average='micro'),4)
    print('F-1 score of model with n-gram range of {}: {}'.format((1,N), score))
```

Figure 4. 9 Implementation of feature extractions

4.7.4 Implementation of Model and Evaluation Metrics

As we discussed in section 4.6, we evaluated the models by using the common machine learning model evaluation metrics like confusion matrix, accuracy, recall, precision, and f1-score.in addition to this, we also used a 10-fold CV.

```
print('accuracy %s' % accuracy_score(y_pred, y_test))
print(classification_report(y_test, y_pred))
#perform k-fold cross validation
from sklearn.model_selection import cross_val_score |
from sklearn.model_selection import ShuffleSplit,cross_val_score
shuffle_split=ShuffleSplit(test_size=0.2,train_size=0.8,n_splits=10,random_state=42)
scores=cross_val_score(wsd_bn,X,y,cv=shuffle_split)
print("cross Validation scores:10-fold ShuffleSplit{}".format(scores))
print("Average Cross Validation score of 10-fold ShuffleSplit:{}".format(scores.mean()))
from sklearn.model_selection import ShuffleSplit,cross_val_score
shuffle_split=ShuffleSplit(test_size=0.2,train_size=0.8,n_splits=5,random_state=42)
scores=cross_val_score(wsd_bn,X,y,cv=shuffle_split)
print("cross Validation scores:5-fold ShuffleSplit{}".format(scores))
print("Average Cross Validation score of 5-fold ShuffleSplit:{}".format(scores.mean()))
import seaborn as sns
from sklearn.metrics import confusion_matrix,accuracy_score
confusion_matrix=pd.crosstab(y_pred,y_test, rownames=['predicted'],colnames=['actual'])
sns.heatmap(confusion_matrix,annot=True,cmap=plt.cm.gray)
print("precision_score:",precision_score(y_test, y_pred, average='weighted'))
print("recall_score:",recall_score(y_test, y_pred, average='weighted'))
print("f1-score:",f1_score(y_test, y_pred, average='weighted'))
print('accuracy %s' % accuracy_score(y_pred, y_test))
```

Figure 4. 10 Implementation of models and evaluation metrics

4.7.5 Implementation of testing WSD model with unseen data

We tested the WSD model with unseen data after the model trained the data. For this purpose, we use the Jupyter Notebook with Python code to predict the sense of the given sentence. The below figure depicts the implementation of the testing phase in the model by taking the sentence from the user and it predicts the sense according to the trained data.

```
# Test the WSD system
test_sentence = "Hagee na'ay Wolayttato doona loyttidi haasayees."
test_tokens = nltk.word_tokenize(test_sentence)
test_features = vectorizer.transform([test_tokens])
predicted_label = classifier.predict(test_features)[0]

# Map the predicted label back to the sense category
predicted_sense = df['Meaning'].astype('category').cat.categories[predicted_label]

print(f"Sentence: {test_sentence}")
print(f"Sense: {predicted_sense}")
```

Sentence: Hagee na'ay Wolayttato doona loyttidi haasayees.
Sense: qaalaa

Figure 4. 11 Implementation of testing model

CHAPTER FIVE

5. RESULT AND DISCUSSION

5.1. Overview

After discussing the principles for implementation and the specifics of this work, the following chapter covers the findings gained from the suggested solution and offers a comparative description of the outcomes using graphs and tables.

5.2. Result Discussion

In this section, we tried to discuss the overall results that we got from the Wolaita language WSD implementation. Additionally, it summarizes the outcomes of each experiment using various data that have been changed and various algorithms.

5.2.1. Result set I: Result of Unsupervised Approaches

Since the unsupervised learning method uses unlabeled datasets, it does not require specialists to label the datasets. Due to this, it consistently performs poorly when measured against other learning methods, including supervised and semi-supervised learning methods. Lack of annotated datasets in the unsupervised machine learning approach. Unsupervised machine learning methods perform poorly since there aren't enough unlabeled datasets. We used four clustering algorithms for our clustering jobs, including EM, Farthest First, Hierarchical Cluster, and Simple K-Means, as we stated in Section 4.5 to choose the best-performing clustering algorithms for the Wolaita language word sense disambiguation. Last but not least, we found that the EM method outperforms the other chosen clustering techniques on average. In summary, unsupervised ML techniques in WSD give an appropriate and scalable solution to the difficulties of labeled data scarcity, domain adaptability, and sense induction. They allow for the finding of latent senses, the capturing of word associations, and the construction of stronger and more diverse WSD systems. This study is the first to use an unsupervised WSD approach for the Wolaita language and is important for future researchers who want to conduct the study in semi-supervised approaches because the approaches require more unlabeled data than labeled data.

Table 5. 1 Result of Unsupervised approach

Datasets	K-means	FarthestFirst	EM	Hierarchical Cluster
Sintta	56.7	49.3	60.4	47.6
Haytta	63.6	51.4	63.6	54.4
Ogiya	62.4	50.3	65.3	53.4
Naaga	59.6	47.4	64.2	48.4
Aadhdha	62.9	50.2	62.8	55.3
Ayfiya	61.3	51.6	63.6	54.6
Doona	62.8	52.2	66.7	55.3
Average	61.7	50.3	63.4	52.7

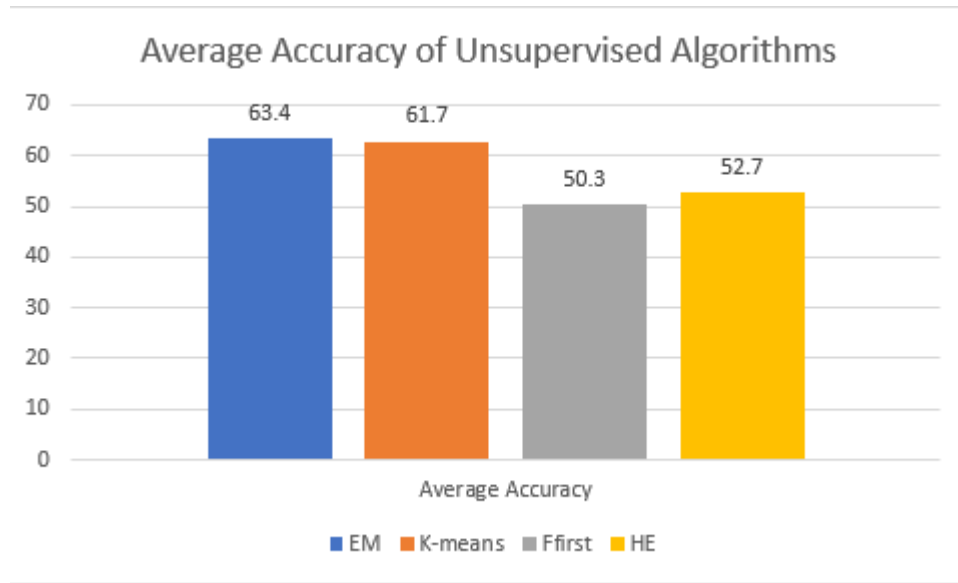


Figure 5. 1 Average accuracy for clustering algorithms

The final results are displayed in the above table and the figure are the best outcomes possible for each ambiguous word using the chosen clustering methods. Because common words may occur often in the chosen datasets, the performance that was obtained by utilizing the chosen clustering algorithms may be low. The more frequently occurring words in the chosen datasets could cause their senses to be incorrectly classified. In general, as shown in the table above, ambiguous words like “Sintta” and “Naaga” have poor performance regarding the selected four clustering algorithms. The reason behind that is those two ambiguous words have five and four

different senses respectively. Therefore, the algorithms cluster those word's datasets into five and four different clusters respectively. These two ambiguous terms may be clustered together by the clustering algorithms together with other words that are difficult to distinguish from their cluster, including more common words. Because of this problem, clustering algorithms perform poorly. Finally, the Expected Maximization clustering algorithm gave the best performance of 63.4 %.

5.2.2. Result Set II: Results of Supervised Approaches

To perform supervised learning, you need annotated training data that the algorithm will use to create a classifier. The fact that it uses a labeled training set composition makes the process easier than it would be with unsupervised learning. But supervised learning typically needs a lot of training data to attain decent results. The below shows the final accuracy result conducted from the selected algorithms.

Table 5. 2 The accuracy result of supervised algorithms

Datasets	NB	AdaBoost	SVM	Bagging	RF	NN
Sintta	79.8	58.8	83.7	78.2	73.7	79.4
Haytta	86.2	61.3	87.7	84.5	79.7	81.8
Ogiya	82.3	65.4	84.4	80.2	76.5	79.5
Naaga	81.6	59.8	83.6	79.6	77.1	76.3
Aadhdha	87.4	60.1	88.6	86.6	82.8	87.4
Ayfiya	80.1	61.8	84.3	79.1	78.1	73.4
Doona	79.8	62.2	83.2	81.7	76.2	72.3
Average	82.4	61.2	85.1	81.4	77.7	78.6

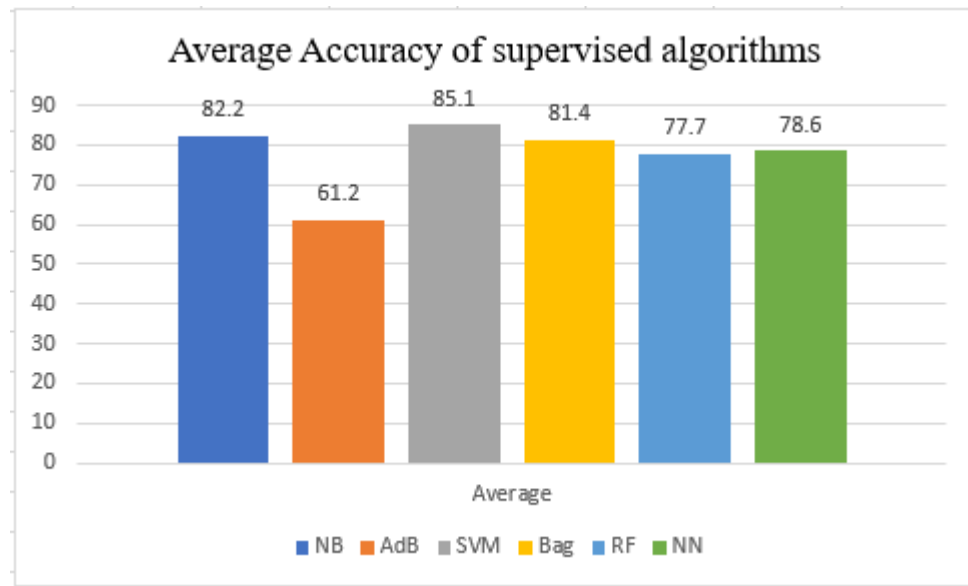


Figure 5. 2 Average Accuracy of Supervised Algorithms

As we have seen from Figure 5.2, among the supervised algorithms selected, the SVM scored a good performance compared to the other five algorithms with an average accuracy of 85.1 from the seven datasets.

5.2.3. Result Set III: Results of Algorithms with Stemming

Stemming can assist reduce the number of distinct word forms in the dataset, which can improve computing performance and minimize the problem of data sparsity. The model has fewer separate elements to learn from since several inflected versions of a word are combined into a single stem. Stemming is generally utilized in natural language processing applications for text standardization and dimensionality reduction. Stemming can assist in assembling related words and minimizing the consequences of data sparsity by breaking down words to their base or root forms. In general, in the context of WSD, stemming is not applied to increase the performance of the model, instead of that it increases the performance by preserving the words to their original form and by utilizing the context information.

Table 5. 3 Results with Stemming

Datasets	The Effect of Stemming on Algorithms											
	Before						After					
	NB	AdB	Bag	SVM	RF	NN	NB	AdB	Bag	SVM	RF	NN
Sintta	79.4	58.8	78.2	83.7	73.7	79.8	80.2	67.3	80.3	85.1	78.6	82.2
Haytta	86.2	61.3	84.5	87.7	79.7	81.8	90.3	75.6	84.3	88.7	81.2	81.2
Ogiya	82.3	65.4	80.2	84.4	76.5	79.5	82.2	70.4	85.1	85.1	80.6	82.6
Naaga	81.6	59.8	79.6	83.6	77.1	76.3	81.8	68.9	84.7	84.5	79.2	84.3
Aadhdha	87.4	60.1	86.6	88.6	82.8	87.4	91.1	74.3	83.2	91.9	82.4	91.1
Ayfiya	80.1	61.8	79.1	84.3	78.1	73.4	81.7	70.4	85.2	84.6	81.7	79.9
Doona	79.8	62.2	81.7	83.2	76.2	72.3	81.2	71.2	83.8	85.7	79.4	81.6
Average	82.4	61.2	81.4	85.1	77.7	78.6	84.1	71.2	83.8	86.5	80.4	83.3

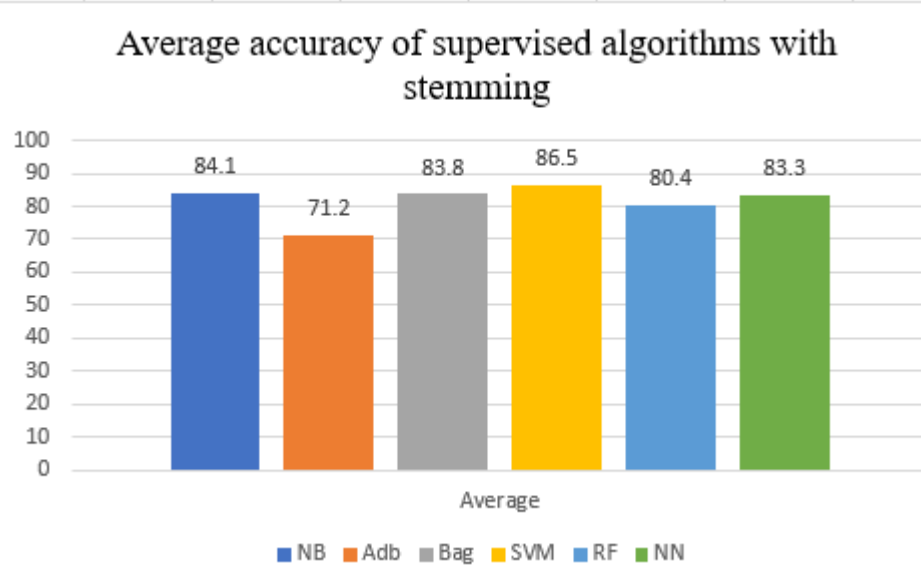


Figure 5. 3 Average Accuracy of supervised algorithms with stemming

As shown in Table 5.2 and Figure 5.2 above, the highest average accuracy of the datasets is 85.1% that of the SMV algorithm, we tried to improve the accuracy by applying one of the preprocessing techniques to the target words only then as we have seen in Table 5.3, we conducted the same SVM algorithm with an accuracy of 86.5%. With this in mind as we have seen in different implemented local languages WSD, the Wolaita language WSD model also improved by using stemming in preprocessing.

5.2.4. Result set IV: The effect of stopwords in algorithms concerning time complexity

Stop words are common words with little significance, such as articles, prepositions, and conjunctions. Removing stop words may minimize the amount of text to be processed, which may result in faster WSD computations. The time complexity of WSD algorithms is often determined by parameters such as the size of the input text, the number of word senses, the disambiguation technique used, and the linguistic resources utilized. Removed stop words can lower the quantity of the input text, potentially improving time complexity. However, the overall influence on temporal complexity will be determined by the individual implementation and the computing steps involved. The Wolaita language also has stopwords and we evaluated the models with and without stopwords. The computational time for all the selected classifiers was evaluated and measured by second (sec).

Table 5. 4 The effect of stopwords concerning time complexity

Datasets	The effect of stop word algorithms concerning time complexity											
	With stop words						After stop words removal					
	NB	AdB	Bag	SVM	RF	NN	NB	AdB	Bag	SVM	RF	NN
Sintta	9.22	10.7	9.71	8.74	13.7	9.87	8.21	8.14	9.52	7.71	11.3	8.82
Haytta	8.32	8.82	7.87	6.65	9.51	9.02	6.22	7.86	6.88	5.72	8.64	8.07
Ogiya	8.02	8.05	8.12	7.04	10.4	8.56	7.01	7.32	7.87	6.94	9.06	7.87
Naaga	8.87	9.87	8.77	7.98	12.4	9.04	7.87	8.12	7.07	7.47	10.8	8.21
Aadhdha	8.11	8.03	8.65	8.01	11.8	7.62	7.11	7.41	7.92	7.32	10.1	7.21
Ayfiya	7.53	8.56	9.61	7.77	14.4	7.87	7.32	6.88	8.71	6.94	12.5	7.03
Doona	7.87	7.96	8.05	7.34	13.6	8.12	7.16	7.11	7.56	6.02	11.4	7.63
Average	8.27	8.86	8.68	7.65	12.3	8.58	7.27	7.55	7.93	6.77	10.5	7.83

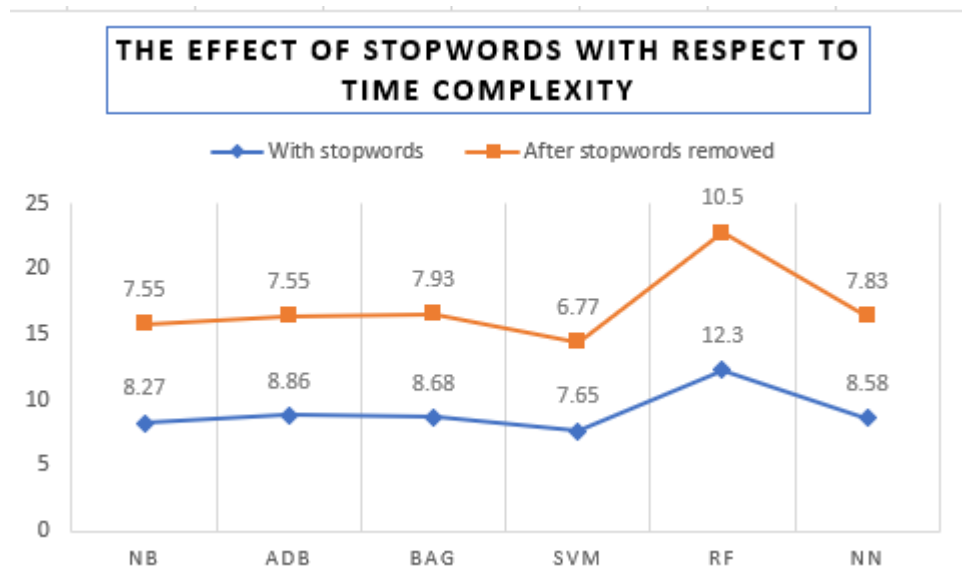


Figure 5. 4 The average stopword effect on computational time

As shown above in Table 5.4 and Figure 5.4, when stopwords were removed from the data, the classifiers' computation time was reduced. As (Ladani & Desai, 2020) stated in their study, some of the advantages of stopword removal include a 35-45% reduction in corpus size, improved efficiency and accuracy of text mining applications, and a reduction in total application time and space complexity. In general, we conclude that stopwords removal reduces the size of the corpus, enhances the performance of algorithms within all the major evaluation techniques (accuracy, precision, recall, and f1-score) value, and decreases space and time complexity as we have seen from implementation as well as from the above-listed study.

5.2.5. Result set IV: Average accuracy of selected feature extraction techniques

As discussed in previous sections, we used four feature extraction techniques in this study. We used the three selected feature extractions Bag of Words, Word2vec, and Tf-Idf integrating with N-gram. Integrating TF-IDF with N-gram features makes use of the benefits of both techniques in WSD. TF-IDF illustrates the significance of terms within a larger document collection, whereas N-grams concentrate on the immediate context surrounding the target word. This integration can improve the representation of the context of the target word and lead to better feature extraction for WSD, thus enhancing disambiguation accuracy. The table and figure below illustrate the average accuracy value of the techniques.

Table 5. 5 Average accuracy of feature extraction techniques

Algorithms	SVM			NB			NN			AdaBoost			Bagging			RF		
	BOW	Word2Vec	TF-IDF & N-gram	BOW	Word2Vec	TF-IDF & N-gram	BOW	Word2Vec	TF-IDF & N-gram	BOW	Word2Vec	TF-IDF & N-gram	BOW	Word2Vec	TF-IDF & N-gram	BOW	Word2Vec	TF-IDF & N-gram
Sintta	76.3	70.2	85.1	80.3	75.8	80.2	77.3	70.4	82.2	59.4	52.8	67.3	78.2	73.5	80.3	73.7	67.4	78.6
Haytta	81.4	72.3	88.7	85.3	76.8	90.3	80.2	76.2	81.2	67.4	60.3	75.6	84.5	77.6	84.3	72.7	70.4	81.2
Ogiya	79.4	70.4	85.1	82.1	80.4	82.2	81.3	80.1	82.6	64.5	62.1	70.4	80.2	74.4	85.1	76.5	78.2	80.6
Naaga	78.3	71.7	84.5	81.7	79.6	81.8	79.8	73.4	84.3	74.5	57.8	68.9	79.6	79.2	84.7	77.1	68.3	79.2
Aadhdha	78.8	74.2	91.9	89.2	81.3	91.1	80.3	71.1	91.1	68.4	61.1	74.3	86.6	77.2	83.2	84.8	71.4	82.4
Ayfiya	82.2	75.3	84.6	81.2	76.4	81.7	78.7	72.4	79.9	67.3	62.8	70.4	79.1	78.6	85.2	73.1	79.4	81.7
Doona	84.3	70.8	85.7	81.8	78.2	81.2	80.6	71.3	81.6	66.2	61.6	71.2	81.7	76.4	83.8	76.2	75.6	79.4
Average	80.1	72.1	86.5	82.9	78.4	84.1	79.7	73.6	83.3	66.8	59.8	71.2	81.4	76.7	83.8	76.3	72.9	80.4

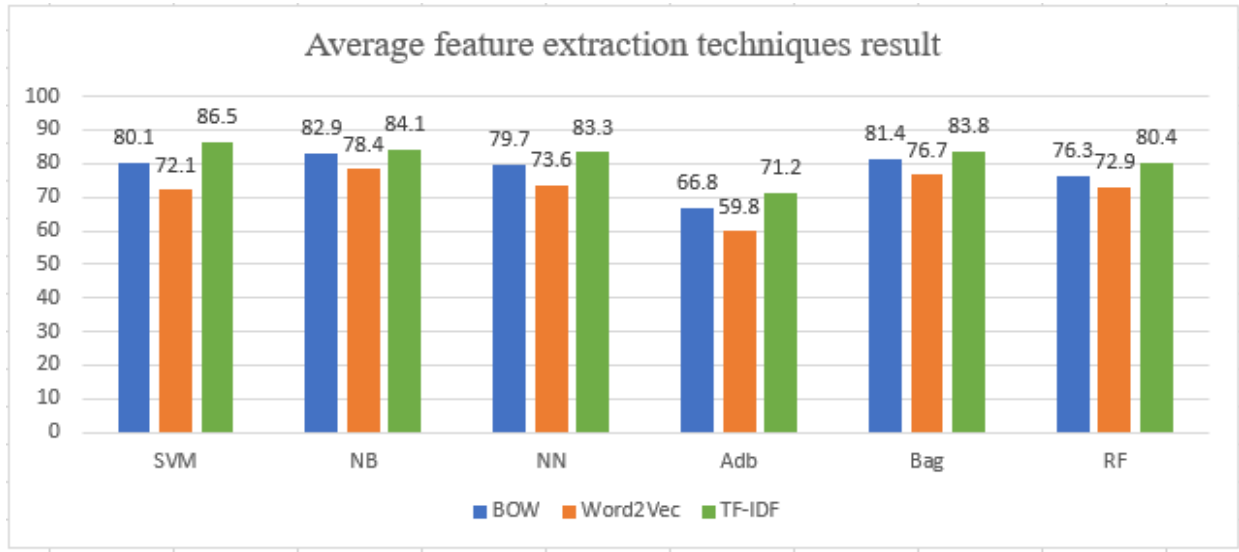


Figure 5. 5 Average accuracy result of feature extraction techniques

As the above results of feature extraction techniques Table 5.5 describe, from the selected techniques TF-IDF with N-gram achieved the maximum accuracy compared other two techniques. As The purpose of n-gram-based WSD is to determine the sense of a given ambiguous word that has the highest joint probability with co-occurring terms in the training dataset and the TF-IDF transformer uses it as the input feature data to transform it to vector form. As (SUHASINI V, 2021) stated, the TF-IDF with N-gram had the highest accuracy in the classifiers like SVC and NB as n-gram splits the given corpus into the given n-gram (unigram,

bi-gram, tri-gram), and the Tf-idf makes the current vector available and stores it in the feature dataset to determine the density of the test word.

5.2.6 Result set IV: Optimum window size for SVM and NB algorithms.

Different researchers have used various WSD techniques for various languages in their research to determine the optimal context window size. To determine the ideal context window size for this language using various methods, WSD researchers were carried out for other local languages like Amharic, Afaan-Oromo, Tigrigna, and geez languages starting from one-one to ten-ten window sizes. But in this study, we conducted WSD for the Wolaita language in selected seven ambiguous words optimal window context for two best-performed algorithms SVM and NB.

Table 5. 6 Window size experimentation result using SVM algorithm

W-size	Sintta	Haytta	Ayfiya	Doona	Naaga	Aadhdha	Ogiya	Average
1-1	82.4	85.3	83.6	86.6	83.2	84.3	85.8	84.4
2-2	81.4	80.2	80.8	83.4	84.3	82.2	83.7	82.2
3-3	80.3	80.8	81.4	82.9	84.5	82.6	81.2	81.8
4-4	80.3	81.2	81.3	80.6	81.6	82.4	80.4	81.1
5-5	88.7	91.9	84.6	85.7	84.5	85.1	85.1	86.5
6-6	81.8	81.6	79.6	83.6	77.2	76.3	81.8	80.3
7-7	72.6	79.4	78.2	83.7	80.8	79.8	80.2	79.2
8-8	82.4	84.3	80.6	86.6	81.2	83.3	85.8	83.4
9-9	81.2	79.8	81.7	83.2	76.3	72.3	81.2	79.4
10-10	80.8	78.2	79.3	80.6	78.6	77.4	81.4	79.5

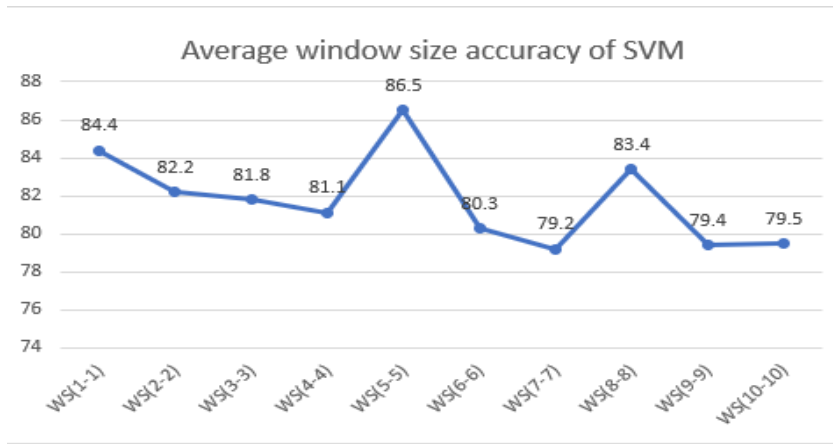


Figure 5. 6 Average window size accuracy result in SVM

As shown in Table 5.6 and Figure 5.6, the optimum window size for the seven datasets is calculated and from the average of the overall datasets window size WS (1-1) and WS (5-5) give the highest accuracy of 84.4% and 86.5% respectively. As (Almuzaini & Azmi, 2020) stated in their study, the best window size for most languages window size is five. With this in mind, our study also gave the best performance on this window size. We also attempted to determine the effect of maximum window size on computational time for SVM. According to (Jurafsky & Martin, 2022), the maximum window size takes a long period to classify the words to the classifier. Furthermore, smaller window sizes produce syntactic meaning rather than semantic meaning. So, in this study, we chose neither maximum nor minimum window sizes.

Table 5. 7 Window size result by computation time in SVM

Window sizes	Window size by computational time cost (sec)							
	Sintta	Haytta	Ayfiya	Doona	Naaga	Aadhdha	Ogiya	Average
1-1	4.23	3.56	3.64	3.31	4.14	3.82	3.45	3.74
2-2	5.43	4.23	4.14	4.73	6.31	5.69	4.43	4.99
3-3	6.34	5.52	5.92	4.97	6.98	6.88	5.22	5.97
4-4	7.53	5.54	6.81	6.18	7.45	7.06	6.43	6.71
5-5	7.71	5.72	6.94	6.02	7.47	7.32	6.89	6.77
6-6	9.52	6.22	8.97	8.19	8.72	8.53	7.01	8.31
7-7	11.4	6.34	9.02	8.54	10.4	8.82	7.72	8.93
8-8	11.1	7.04	9.53	8.86	10.9	10.1	7.44	8.96
9-9	12.6	8.33	9.68	8.92	11.2	10.5	8.73	9.99
10-10	13	8.78	9.87	9.08	11.6	11	9.92	10.5

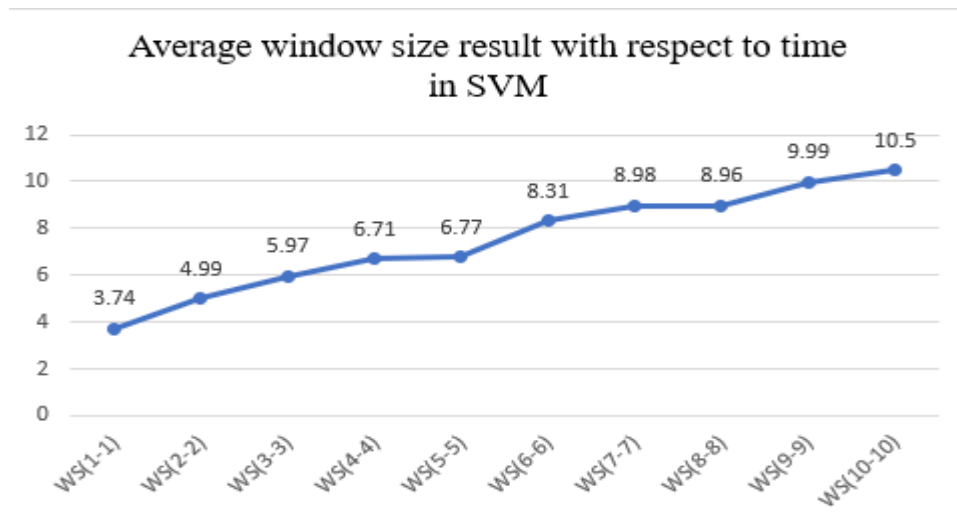


Figure 5. 7 Average window size by computational time in SVM

As we have seen from Figure 5.7, if the window size increases, the time cost of the classifier also increases. This illustrates that if the quantity of data increases, the maximum window sizes have the worst computing time. As (WEBB et al., 2002) stated, several models were created to help users' needs via the Internet.

The amount of data processed as well as the number of users has been growing. As a result, analyzing the models in terms of processing time boosts the validity of the model.

Table 5. 8 Window size experimentation result using NB algorithm

W-size	Sintta	Haytta	Ayfiya	Doona	Naaga	Aadhdha	Ogiya	Average
1-1	80.6	82.4	84.3	86.6	81.2	83.3	85.8	83.4
2-2	81.4	80.3	81.8	82.9	83.5	82.6	81.2	81.7
3-3	83.6	81.3	79.8	80.7	78.6	84.5	83.4	81.7
4-4	79.6	81.8	79.6	80.6	77.2	76.3	79.8	79.3
5-5	82.2	90.3	91.1	81.8	81.7	81.2	84.1	84.1
6-6	80.2	83.7	80.8	83.4	84.3	82.2	81.4	82.2
7-7	78.9	79.4	81.2	82.3	77.7	80.6	83.7	80.5
8-8	79.3	79.8	83.4	80.4	81.4	82.2	80.6	81.1
9-9	81.7	81.2	80.8	83.2	78.3	77.3	81.2	80.6
10-10	79.3	78.8	79.2	79.6	76.6	77.4	80.4	79.9

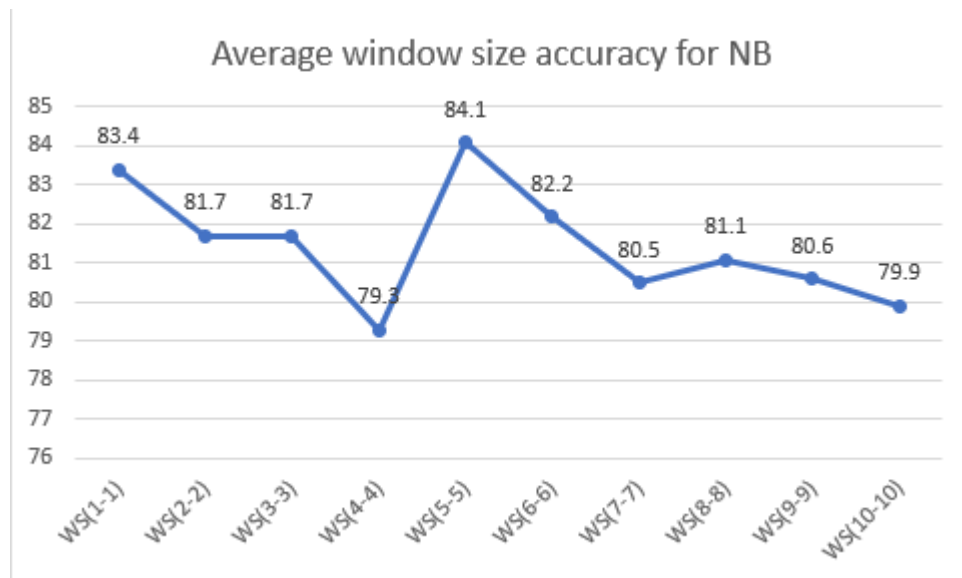


Figure 5. 8 Average window size accuracy in NB

As we have seen in above Table 5.8 and Figure 5.8, the average accuracy of the NB algorithm has a good performance at WS (1-1) with an accuracy of 83.4%.

Therefore, the window size of 1-1 is enough to distinguish the senses between the Wolaita language datasets in the NB algorithm. Generally, from all selected algorithms the average highest accuracy rate is at window size WS (5-5).

Table 5. 9 Selected evaluation metrics with SVM

Datasets	Precision	Recall	Accuracy	F1-score
Sintta	78.2	80.3	85.1	79.4
Haytta	84.5	85.3	88.7	86.2
Ogiya	80.2	82.1	85.1	82.3
Naaga	79.6	81.7	84.5	81.6
Aadhdha	86.6	89.2	91.9	87.4
Ayfiya	79.1	81.2	84.6	80.1
Doona	81.7	81.8	85.7	79.8
Average	81.4	82.9	86.5	82.4

As it is shown in the table above Table 5.9, the average accuracy of SVM is 86.5% and F1-score is 82.4%.

Table 5. 10 Selected evaluation metrics with NB

Datasets	Precision	Recall	Accuracy	F1-score
Sintta	79.8	80.3	80.2	78.2
Haytta	81.8	85.3	90.3	84.5
Ogiya	79.5	82.1	82.2	81.2
Naaga	76.3	81.7	81.8	80.6
Aadhdha	87.4	89.2	91.1	85.6
Ayfiya	73.4	81.2	81.7	79.1
Doona	72.3	81.8	81.2	80.7
Average	78.6	82.9	84.1	81.4

As it is shown in the table above Table 5.10, the average accuracy of NB is 84.1% and F1-score is 81.4%.

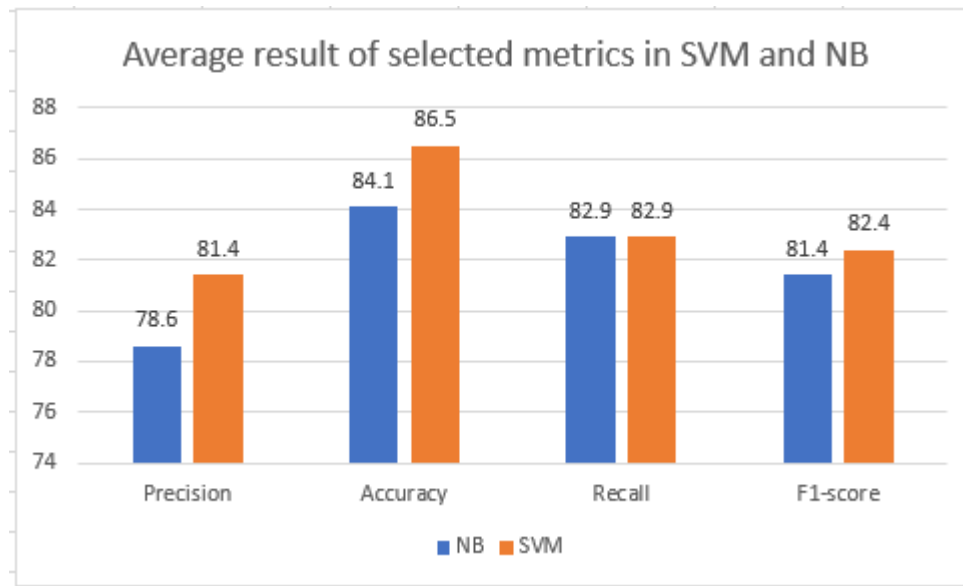


Figure 5. 9 Evaluation metrics average result in SVM and NB

The above Figure 5.9, describes the four different evaluation metric average value in the models, and among the metrics, accuracy had a good result in both SVM and NB algorithms.

5.2.7. Result set V: Confusion matrix Results in SVC

This result section describes the confusion matrix result according to the dataset we added in the study of Tamirat (Temesgen, 2021), which are the ‘Sintta’, and ‘Haytta’ datasets. According to (Almuzaini & Azmi, 2020), the feature extraction aimed the most window size is WS11 (5-5). Taking their study, we also evaluated the confusion matrix by window size 5.

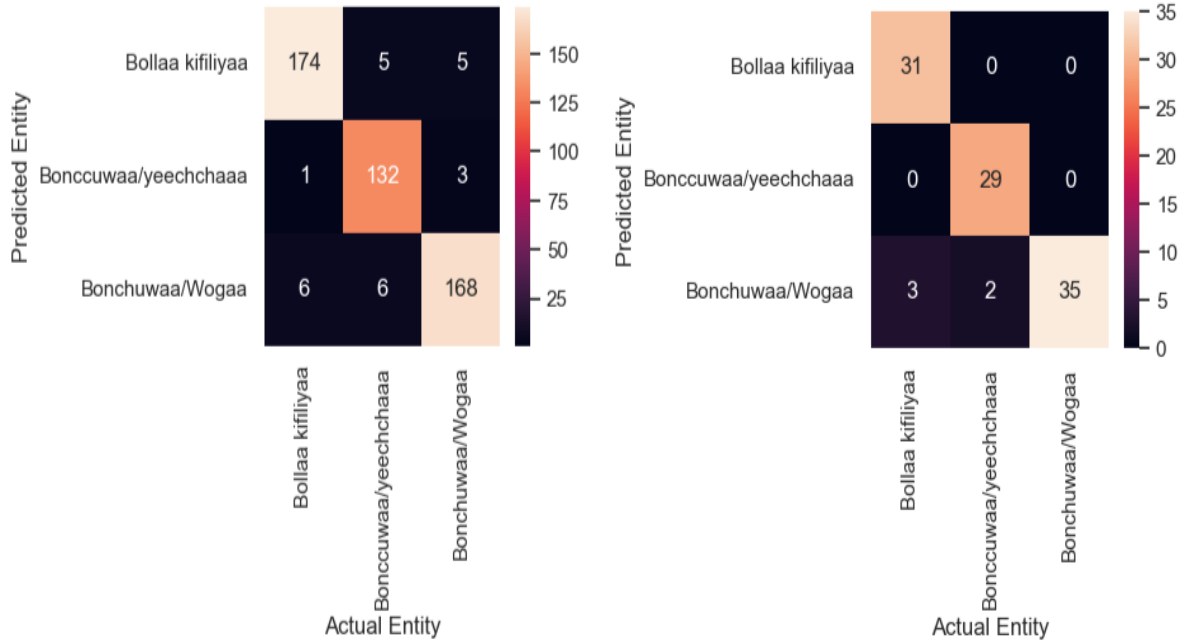


Figure 5.10 Confusion matrix result in train and test data for the 'Haytta' dataset

As we can see from the above figures, we evaluated the confusion matrix result for both train and test data in all the classifiers applied in our study. While, among the algorithms as shown in the above tables, the SVC algorithm gave the best performance. Putting this in our mind, we discussed the confusion matrix for the two datasets listed above. As described in Figure 5.10 the left-side figure shows the train data result for dataset 'Haytta', the SVC classifiers classified 474 words correctly and 26 sense words incorrectly among the overall 500 sense words and the right-side figure shows the test data result for the same dataset with the classifier classified 95 sense examples correctly and 5 sense examples incorrectly out of 100 sense examples.

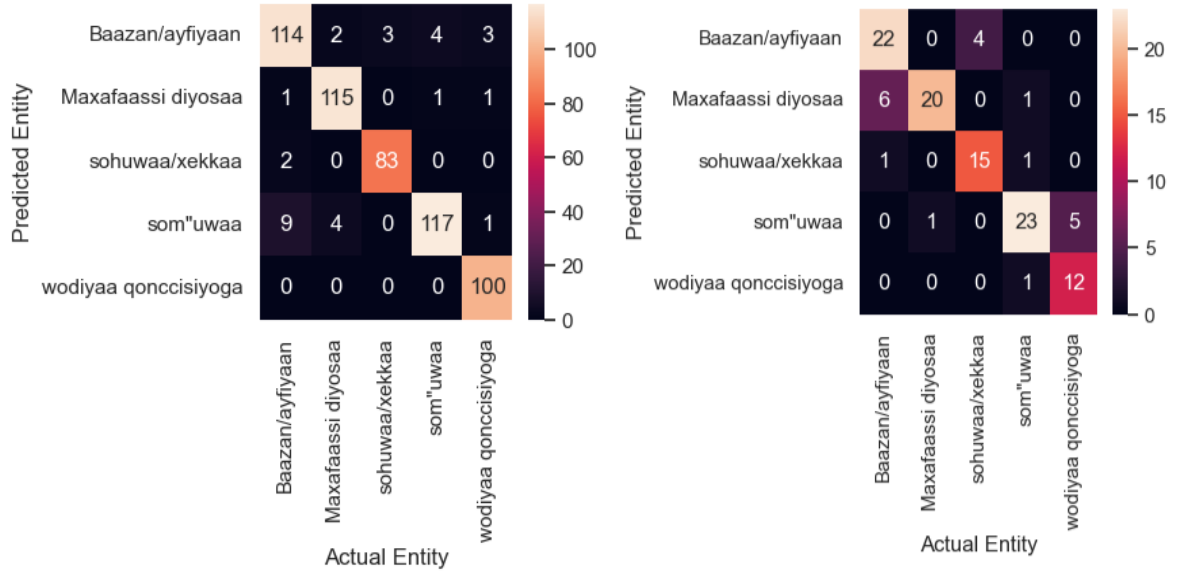


Figure 5.11 Confusion matrix result for the 'Sintta' dataset

As shown in the above figures, we evaluated the confusion matrix result for both train and test data in also for dataset 'Sintta'. As described in Figure 5.11 the left-side figure shows the train data result for dataset 'Sintta', the SVC classifiers classified 529 words correctly and 31 sense words incorrectly among the overall 560 sense words and the right-side figure shows the test data result for the same dataset with the classifier classified 92 sense examples correctly and 20 sense examples incorrectly out of 112 sense examples.

5.2.8. Result set VI: Average cross-validation (CV) Results in SVM and NB

Cross-validation is also one of the popular evaluation methods for machine learning algorithms. We used 10-fold CVs from K-fold CVs, 9 folds were used for training, and 1-fold was used for testing, which means the first nine folds were simultaneously for training data evaluation and the rest K-1 fold is for the test data evaluation.

Table 5.11 Average CV result in SVM and NB

Datasets	SVM	NB
Sintta	0.893	0.864
Haytta	0.874	0.873
Naaga	0.837	0.857
Ogiya	0.824	0.863
Ayfiya	0.868	0.823
Aadhdha	0.844	0.838
Doona	0.852	0.841

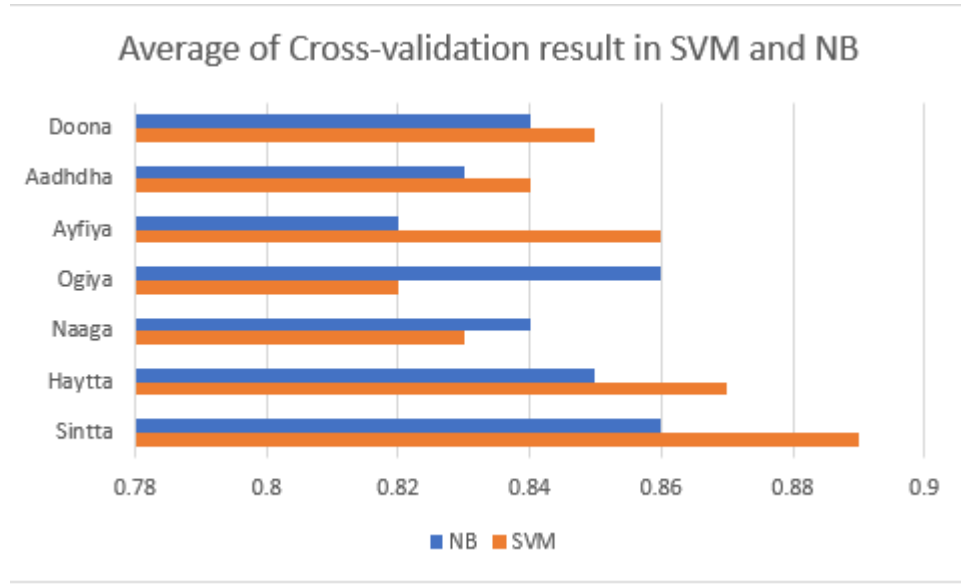


Figure 5. 12 Average Cross-validation result in SVM and NB

5.2.9. Statistically significance test for algorithms

Statistical significance suggests that the observed differences in performance between the algorithms are unlikely to have occurred by chance alone. This finding provides evidence that there are meaningful distinctions in the effectiveness of the algorithms for the specific task we investigated. With this in mind, we performed an analysis of the algorithms based on the accuracy performance they achieved after stemming. To do this we used **t-test** techniques to check whether the difference between the algorithms is statistically significant or not. For this purpose, first, we evaluated the probability value (p-value) by importing '**scipy. stats**' from the statistical Python libraries and using the '**numpy**' for data calculation and manipulation. The t-test is performed using the **ttest_rel()** function from the **scipy. stats** module. The p-value is then compared to the significant level (alpha) of 0.05 and if the $p\text{-value} < 0.05$ the difference between the two algorithms is statistically significant, otherwise the difference is not statistically significant. We evaluated the p-value for five other algorithms with the highly performed algorithm which is SVM using the above Python libraries.

Table 5. 12 Probability value (p-value) of five algorithms with SVM

Algorithms	p-value
NB	0.023
Bagging	1.68e-05
Neural Network	0.013
AdaBoost	0.00025
Random Forest	0.00013

By demonstrating statistical significance, our research contributes to understanding which algorithms are more suitable and effective for WSD in the Wolaita language. This information can be valuable for practitioners, researchers, and developers working in the field of natural language processing or WSD in similar contexts.

5.3 Discussion

As we described in the above figures and tables, the overall experiment result that is essential to fulfill the specific objectives and to solve the research questions. In this section, we discussed the results and the answers to the research questions.

RQ1: Which machine learning algorithm is most effective for WSD model development?

Ans: From the selected six classification and bootstrapping algorithms in supervised learning the SVM and Naïve Bayes algorithm give the best performance compared to the other four algorithms (AdaBoost, Neural Network, Bagging, and Random Forest). In all of the seven datasets, the SVM algorithm gives the highest accuracy of **86.5%** with stemming in the preprocessing. When in the study of Temesgen (Temesgen, 2021), they achieved an accuracy of 83.22%, so in our study, we improved the performance of the algorithm by using N-gram as a feature, by changing the kernel of the SVM classifier and applying stemming technique in preprocessing. In the case of an unsupervised approach, the EM algorithm performs better when compared to the other three clustering algorithms. After implementation, we have got the accuracy of EM is 63.4 % and the least performed algorithm is FarthestFast, with an accuracy of 50.3%.

RQ2: Which features most contribute to enhancing the existing WSD model?

Ans: In the recent Wolaita language WSD, as the feature selection and extraction they used BOW, Tf-idf, and Word2vec. In our study in addition to those features, we used Tf-idf with N-gram to improve the performance of the model as the n-gram finds the co-occurrence words of target ambiguous words and is an efficient compression of large amounts of the corpus, as it states how often each sequence of words occurs. From the selected three feature extraction Tf-idf with n-gram performed well over other methods. As we discussed above in the result sections the best-performed algorithm is SVM, in the SVC there is a major feature called the kernel. The kernel is an independent variable in the classifier and in the model, we used the kernel type which is 'linear'. So, the linear kernel is good for the performance of SVM in our study. As discussed in section 3.5, we used optimization named SGD techniques to improve the performance of the models by decreasing the degree of error. When using optimization techniques, the models may face overfitting of data, to overcome this challenge we used L2 regularization to avoid overfitting.

RQ3: To what extent does stemming affect the performance of the WSD model?

Ans: Stemming has been found to significantly increase the performance of WSD for morphologically complicated languages, as stated in previous chapters (Lovins, 2020). In this study, we tried to apply stemming for the target words in preprocessing stages and as we have seen in the above tables which is the accuracy of the models, the data with stemming give the best performance compared to the data without stemming. Not only in the Wolaita language WSD stemming improves the performance of the datasets, but also in other local languages like, Afaan-Oromo, Amharic, and Tigrigna languages also stemming improved the performance in WSD (Assemu, 2021), (Feyisa, 2015), (Teklay, 2018).

CHAPTER SIX

6. CONCLUSION AND RECOMMENDATIONS

6.1. Conclusion

In natural language, many words have multiple meanings, and each one is influenced by their context. Word Sense Disambiguation (WSD) is a term for the automated procedure that identifies word senses about one another. A significant subtask in many NLP applications, including machine translation (MT), semantic mapping (SM), semantic annotation (SA), and ontology learning (OL), is the challenge of differentiating between several alternative meanings of a word. Additionally, numerous applications, including information retrieval (IR), information extraction (IE), speech recognition (SR), and others, are thought to benefit from its enhancement.

To assign senses to words in a given context there are three basic approaches in NLP, that are corpus-based, knowledge-based, and hybrid approaches. Among the three approaches, we choose the corpus-based approach for this study. Corpus-based approaches by itself classified into three supervised, unsupervised, and semi-supervised. In this study, we mainly focused on the supervised learning methods and we tried to implement the unsupervised approaches by using four different algorithms from clustering algorithms. In this study for the first time, we conducted the clustering algorithms for the Wolaita language WSD, as there is no prior research done research for unlabeled data on the Wolaita language. For the reason of time constraints, we didn't conduct the ambiguous word data for the Wolaita language in semi-supervised approaches and we left it to the other researcher who want to exalt their study on the Wolaita language WSD.

As the Wolaita language is one of the local languages which is not prior produced large-size data, we prepared the corpus manually for this study. Although the Wolaita language is one of the morphologically rich languages, we selected only seven ambiguous words from the Wolaita language dictionary. These words are, Sintta, Haytta, Ogiya, Naaga, Ayfiya, Aadhdha, and Doona. For those selected words we collected the data from several sources to make our data divergent. For the seven words we prepared seven different datasets and the total data we used for this study is 3560 sentences including the ambiguous words. And then we have done different preprocessing techniques to make the data readable form to the models.

In the preprocessing phase, we also applied stemming to ambiguous words and this is never done in previous studies of the Wolaita language WSD. For future extraction, we used N-gram with Tf-idf in this study in addition to BOW and Word2vec. We used n-gram because Bow and Word2vec have some drawbacks, mainly the words which are in the context are not ordered as they are collected, so there is a loss of context from the text. So, to get around these limitations, we used n-grams because they deal with the problems by establishing an order that keeps the context. finally, the n-gram with Tf-idf became the best in the support vector classifier (SVC).

For unsupervised approaches, we selected four clustering algorithms Expectation Maximization (EM), Simple k-means (K-means), Farthest First, and Hierarchical Clusters. We selected these algorithms because in other local languages like Afaan-Oromo, and Tigrigna language they perform well. Finally, after the experimentation, the EM algorithm gave an effective performance compared to the other three clustering algorithms in this study with **63.4%**.

For supervised approaches, we, selected six different boosting and classification algorithms like Support Vector Machine (SVM), Naïve Bayes (NB), AdaBoost (AdB), Bagging (Bag), Neural Network (NN), and Random Forest (RF), among these algorithms SVM and Naïve Bayes achieved the best performance **86.5%** and **84.1%** respectively after stemming applied.

6.2. Recommendation and Future Works

Research on word sense disambiguation needs several linguistic tools, including a thesaurus, WordNet, machine-readable dictionaries, and an efficient Wolaita language stemmer. This was a huge issue for us because the Wolaita language doesn't have access to these facilities. Another issue that forced us to restrict our analysis to seven ambiguous words in the language was the absence of sense-annotated data for the language. Insufficient annotated data also forces us to restrict the number of sentences in our sample to 3560. As a result, we have the following suggestions, which also contain future study directions for WSD of texts in the Wolaita language:

- 🚧 This work only focused on modeling WSD to address lexical ambiguity. More research is needed to address various types of ambiguity in the Wolaita language, such as semantic, phonological, and referential ambiguities.

- ✚ In addition to the corpus-based approach, knowledge-based and hybrid approaches based on WSD have also been implemented for other local languages. These approaches are recommended to investigate the Wolaita language WSD as well.
- ✚ Language resources like the thesaurus, lexicon, wordnet, and machine-readable dictionaries, are developed for other languages by different researchers. Researchers were unable to conduct a study on the WSD of the Wolaita language since such materials for the language were unavailable. Other researchers can undertake a study to prepare these resources with involved institutions or people by taking into account their contribution to WSD.
- ✚ With time constraints, we didn't develop a prototype in this study. So, we recommend that future researchers develop the prototype for Wolaita Language WSD with any of the approaches they going to use.

=====

SPECIAL ACKNOWLEDGMENT

This research project was funded by Adama Science and Technology University under grant number: ASTU/SM-R/833/23

Adama, Ethiopia

REFERENCES

- Agirre, E., & Edmonds, P. (2007). *Word Sense Disambiguation Algorithms and Applications*. www.springer.com.
- Agirre, E., & Martinez, D. (2001). *Knowledge Sources for Word Sense Disambiguation*.
- Almuzaini, H. A., & Azmi, A. M. (2020). Impact of Stemming and Word Embedding on Deep Learning-Based Arabic Text Categorization. *IEEE Access*, 8, 127913–127928. <https://doi.org/10.1109/ACCESS.2020.3009217>
- Amlakie, A. (2020). *Corpus Based Word Sense Disambiguation for Geez Language*.
- Bala, R., Sikka, S., & Singh, J. (2014). A Comparative Analysis of Clustering Algorithms. In *International Journal of Computer Applications* (Vol. 100, Issue 15). <http://facweb.cs.depaul.edu/mobasher/classes/ect584/weka/p>
- Banerjee, S., & Pedersen, T. (2002). *An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet*. <http://www.d.umn.edu/~bane0025,~tpederse>
- Beldados, D. (2013). *AUTOMATIC THESAURUS CONSTRUCTION FROM WOLAYTTA TEXT*.
- Birhane, T. (2017). *Afaan Oromo Word Sense Disambiguation Using WordNet*.
- Borah, P. P., Talukdar, G., & Baruah, A. (2014). Approaches for Word Sense Disambiguation-A Survey. In *International Journal of Recent Technology and Engineering (IJRTE)* (Issue 3). <https://www.researchgate.net/publication/265254449>
- Buscaldi, D., Rosso, P., Pla, F., Segarra, E., & Arnal, E. S. (2006). Verb sense disambiguation using support vector machines: Impact of WordNet-extracted features. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3878 LNCS, 192–195. https://doi.org/10.1007/11671299_21
- Dorr, B. (1994). *Machine translation divergences: A formal description and proposed*.
- Dureti, S. (2017). *A Generic Approach towards All Words Amharic Word Sense Disambiguation*.
- Ermias, N. (2021). *FAKE NEWS DETECTION FOR AMHARIC LANGUAGE USING DEEP LEARNING*.
- Faisal, E., Nurifan, F., & Sarno, R. (2018). *Word Sense Disambiguation in Bahasa Indonesia Using SVM*.
- Ferreira, A., & Figueiredo, M. (n.d.). *Boosting Algorithms: A Review of Methods, Theory, and Applications*.
- Feyisa, G. (2015). *Afaan Oromo Unsupervised Word Sense Disambiguation (AOUWSD)*.
- Gonfa, S. O. (2018). *WORD SENSE DISAMBIGUATION FOR AFAAN OROMO USING KNOWLEDGE BASE*.

- Habtamu, F. (2010). *SPEAKER DEPENDENT SPEECH RECOGNITION FOR WOLAYTTA LANGUAGE*.
- Ide, N., & Veronis, J. (1998). *Introduction to the Special Issue on Word Sense Disambiguation: The State of the Art*.
- Ide, N., & Veronis, J. (1998). *Introduction to the Special Issue on Word Sense Disambiguation: The State of the Art*.
- Jurafsky Martin. (2007). *Speech and Language Processing_ An Introduction to Natural Language Processing*.
- Jurafsky, D., & Martin, J. H. (2022). *Speech and Language Processing An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition Third Edition draft Summary of Contents*.
- Kassie, T. (2009). *WORD SENSE DISAMBIGUATION FOR AMHARIC TEXT RETRIEVAL: A CASE STUDY FOR LEGAL DOCUMENTS*.
- Ladani, D. J., & Desai, N. P. (2020). *Stopword Identification and Removal Techniques on TC and IR applications: A Survey* (6th ed.).
- Lesk, M. (1986). *Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone*.
- Lessa, L. (2003). *DEVELOPMENT OF STEMMING ALGORITHM FOR WOLAYTTA TEXT*.
- Loh, W. Y. (2011). Classification and regression trees. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(1), 14–23. <https://doi.org/10.1002/widm.8>
- Lovins, J. B. (2020). Development of a Stemming Algorithm*. In *Mechanical Translation and Computational Linguistics* (Vol. 11, Issue 2).
- Majumder. (2021). *Word Sense Disambiguation: Importance in Natural Language Processing*. <https://www.pexels.com/photo/book-eyeglasses-eyewear-page-261857/>
- Mante, R., & Kshirsagar, M. (2014). *A Review of Literature On Word Sense Disambiguation*. www.ijcsit.com
- Mara, M. (2018). *ENGLISH-WOLAYTTA MACHINE TRANSLATION USING STATISTICAL APPROACH*.
- Marcot, B. G., & Hanea, A. M. (2021). What is an optimal value of k in k-fold cross-validation in discrete Bayesian network analysis? *Computational Statistics*, 36(3), 2009–2031. <https://doi.org/10.1007/s00180-020-00999-9>
- Mcculloch, W. S., & Pitts, W. (1990). A LOGICAL CALCULUS OF THE IDEAS IMMANENT IN NERVOUS ACTIVITY* n. In *Bulletin of Mothemnticnl Biology* (Vol. 52, Issue 2).
- McEnery, T., & Wilson, A. (2001). *Corpus Linguistics*. 2.

- Miller, G. A., Leacock, C., Teng, R., & Bunker, R. T. (1993). *A SEMANTIC CONCORDANCE*.
- Motomichi, W. (2014). *A Sketch Grammar of Wolaytta*.
- Navigli, R. (2009). *Word Sense Disambiguation: A Survey*, *ACM Computing Surveys*.
- Neima, M. (2022). *Amharic Word Sense Disambiguation Using Transfer learning*. 19–21.
- NG'ANG'A, W. (2005). *Word Sense Disambiguation of Swahili: Extending Swahili Language: Faculty of Arts*.
- Ozdemir, V. (2009). *Word Sense Disambiguation for Turkish Lexical sample*.
- Pinar Y, e. a. (2005). *The effect of Windowing in WSD, ISCiS*.
- Rivest, R. L. (1987). Learning Decision Lists. In *Machine Learning* (Vol. 2).
- Segid, H. (2015). *Amharic Word Sense Disambiguation using WordNet*.
- Sharma, N., Sai, S. S., Litoriya, R., Ww, W.: Bajpai, A., & Litoriya, M. R. (2012). *Comparison of the various clustering algorithms of Weka tools Blockchain View project Fog Computing View Project International Journal of Emerging Technology and Advanced Engineering Comparison the various clustering algorithms of Weka tools* (Vol. 2, Issue 5). www.ijetae.com
- Sharma, R., & Bhatia, P. (2008). *Word Sense Disambiguation for the Hindi Language*.
- Shi, N., Liu, X., & Guan, Y. (2010). Research on k-means clustering algorithm: An improved k-means clustering algorithm. *3rd International Symposium on Intelligent Information Technology and Security Informatics, IITSI 2010*, 63–67. <https://doi.org/10.1109/IITSI.2010.74>
- Solomon, A. (2021). *Unsupervised Machine Learning Approach for Word Sense Disambiguation to Amharic Words*.
- Solomon, M. (2010). *WORD SENSE DISAMBIGUATION FOR AMHARIC TEXT: A MACHINE LEARNING APPROACH*.
- SUHASINI V, D. N. V. (2021). *A Hybrid TF-IDF and N-Grams Based Feature Extraction Approach for Accurate Detection of Fake News on Twitter Data*.
- Teklay, M. (2018). *WORD SENSE DISAMBIGUATION FOR TIGRINA LANGUAGE USING A SEMI-SUPERVISED MACHINE LEARNING APPROACH*.
- Temesgen, T. (2021). *Word Sense Disambiguation for Wolaita Language Using Machine Learning Approach*.
- Tesema, W. (2017). Word Sense Disambiguation and Semantics for Afan Oromo Words Using. *International Journal of Research Studies in Science*, 4–6.
- Tesfa, K. (2013). *Word Sense Disambiguation for Afaan Oromo Language*.

- Toophphiya Hayyo Billaa Woshettaa. (2008). *Unpublished*.
- Vickrey, D., Biewald, L., Teyssier, M., & Koller, D. (n.d.). *Word-Sense Disambiguation for Machine Translation*. <http://www.isi.edu/>
- Wakasa, M. (2008). *A Descriptive Study of the Modern Wolaytta Language*. www.A-PDF.com
- Wassie, G., Ramesh, B., Teferra, S., & Meshesha, M. (2014). A Word Sense Disambiguation Model for Amharic Words using Semi-Supervised Learning Paradigm. *Science, Technology, and Arts Research Journal*, 3(3), 147. <https://doi.org/10.4314/star.v3i3.25>
- WEBB, G. I., PAZZANI, M. J., & BILLSUS, D. (2002). *Machine Learning for User Modeling*.
- Workineh, W. (2020). *Attention-based Amharic-to-Wolaita Neural Machine Translation*.
- Xie, J., & Jiang, S. (2010). A simple and fast algorithm for global K-means clustering. *2nd International Workshop on Education Technology and Computer Science, ETCS 2010*, 2, 36–40. <https://doi.org/10.1109/ETCS.2010.347>
- Yarowsky, D. (1992). *Word-sense disambiguation using statistical models of Roget's categories trained on large corpora*. 454. <https://doi.org/10.3115/992133.992140>
- Yehoshua Bar-Hillel. (2021). Yehoshua Bar-Hillel. In *Wikipedia*.
- Yehuwalashet, B. T. (2016). *Hybrid Word Sense Disambiguation Approach for Afaan Oromo Words*.
- Ying, X. (2019). An Overview of Overfitting and its Solutions. *Journal of Physics: Conference Series*, 1168(2). <https://doi.org/10.1088/1742-6596/1168/2/022022>
- Zhao, Y., & Karypis, G. (2005). Hierarchical clustering algorithms for document datasets. *Data Mining and Knowledge Discovery*, 10(2), 141–168. <https://doi.org/10.1007/s10618-005-0361-3>
- Zhou, X., & Han, H. (2005). *Survey of Word Sense Disambiguation Approaches*. www.aaai.org

APPENDIXES

Appendix I: The selected ambiguous words with their different senses

Words	senses	Sense of the words				
		Sense 1	Sense 2	Sense 3	Sense 4	Sense 5
Doonaa	3	Issiba lanqiyaa	Bollaa kifiliyaa	Qaalaa	-	-
Naaga	4	Iita teqiyooqa	Koyyetyaaba ootiyooqa	Gati ekkiyooga	Higgiya bonchchiyooga	-
Haytta	3	Siyiiya bollaa kifiliyaa	Bonchuchuwa	Bonccuwaa	-	-
Ayfiyaa	3	Xeeliyaa bollaa kifiliyaa	Zerettaa	Murutaa	-	-
Ogiyaa	3	Kaammenne asay biyoosaa	Siraatanne wogaa	Hiillaa	-	-
Sintta	5	Som’’uwaa	Wodiyaa qonccissiyooqa	Sohuwaa/ xekka	Maxaafassi diyoossa	Baazan/ayfiyan
Aahdha	3	Haraakko biyoogaa	Kanttichida wodiyaa	Daridabaa		

Appendix II: Sample dataset format

Neeni	taayo	qanxxido	damoozaa	qoodanaw	sinttappe	yiyo	taayoo	markkatassa									sohuwaa/xekkaa	
			He	gatimaa	sinttan	birasettid	dosona										Baazan/ayfiyaan	
	Taani	ha'i	neeni	yaana	sinttan	maas											wodiya qonccisiyoga	
				Eta	sinttan	maattaa	wottoppa										Baazan/ayfiyaan	
		Wude	ubba	baappe	sinttara	laaggidi	ekkidi	biis									Baazan/ayfiyaan	
			Ne	laggetu	sinttan	koyyadda	ekka										Baazan/ayfiyaan	
		Ekkada	ne	dabbotu	sinttan	wotta											Baazan/ayfiyaan	
Ha	kiitta	taani	ta	goddaa	sinttan	nashettan	mala	kiitas									Baazan/ayfiyaan	
			Ha	dorssa	sinttay	yashshees											som"uwaa	
				A	sinttaa	xeeliyood	mule	iitta	yashshees								som"uwaa	
				Boorassi	sinttay	aaho											som"uwaa	
				Gelleshuv	sinttay	iittees											som"uwaa	
			Mittay	na'assi	sinttaa	puuziis											som"uwaa	
			Ha	ne	sinttan	de'iyaa	meehe	ubbay	oosse								Baazan/ayfiyaan	
Hegaa	gishshaw	bagga	immiyob	baappe	sinttawu	yeddiis											wodiya qonccisiyoga	
				Hagappe	sinttawu	ne	sunttay	yaqooba									wodiya qonccisiyoga	
				Woochiyo	keelaa	sinttaa	19-24n	Bassa	de'uwan	hanidabay	de'ees						Maxafaassi diyosaa	
				Huuphe	yohuwaa	sinttaa											Maxafaassi diyosaa	
				gliigiddida	sinttaa												Maxafaassi diyosaa	
Laappunn	2008	nu	kawotetta	haggazuu	sinttaa	5	xeella										Maxafaassi diyosaa	
	Haggazo	timirtte	keettaa	maxaafaa	sinttaa	128	de'iyaga	haasayan	shiishshiyogaa								Maxafaassi diyosaa	

Appendix III: Stopwords of the Wolaita Language

Awde	aybaassi	ookko
wookku	aybaayoo	oow
aybee	ooro	oossi
aybaa	iiro	oogey
aybaaw	neegeta	aageta
taageta	inteegeta	aako
eetaa	aaro	iiro
nuuro	hintero	eetaaro
an	neero	baga
guutta	aa	ne
aw	ikko	asati
nu	taw	uu
entara	ba	daroo
eeti	mala	nukko
yaara	kuma	hegati
taani	ubba	yoota
hagani	tani	hinte
yiisi	bitanney	hagata
yaan	nura	iira
haan	beta	iiko

Appendix IV: Data annotator's result using Fleiss kappa

Fleiss Multirater Kappa

[DataSet0]

Overall Agreement^a

	Kappa	Standard Error	Asymptotic z	Sig.	Asymptotic 95% Confidence Interval	
					Lower Bound	Upper Bound
Overall Agreement	-.084	.039	-.879	.379	-.111	.042

a. Sample data contains 7 effective subjects and 5 raters.

Agreement on Individual Categories^a

Rating Category	Conditional Probability	Kappa	Standard Error	Asymptotic z	Sig.	Asymptotic 95% Confidence ... Lower Bound
.00	.227	-.127	.120	-1.062	.288	-.361
110.00	.000	-.029	.120	-.246	.806	-.264
111.00	.000	-.029	.120	-.246	.806	-.264
112.00	.000	-.029	.120	-.246	.806	-.264
113.00	.000	-.029	.120	-.246	.806	-.264
114.00	.000	-.029	.120	-.246	.806	-.264
122.00	.000	-.029	.120	-.246	.806	-.264
126.00	.000	-.029	.120	-.246	.806	-.264
130.00	.000	-.029	.120	-.246	.806	-.264
132.00	.000	-.029	.120	-.246	.806	-.264
140.00	.000	-.029	.120	-.246	.806	-.264
150.00	.000	-.029	.120	-.246	.806	-.264
159.00	.000	-.029	.120	-.246	.806	-.264
164.00	.250	.153	.120	1.282	.200	-.081
165.00	.000	-.029	.120	-.246	.806	-.264
167.00	.000	-.029	.120	-.246	.806	-.264
170.00	.000	-.029	.120	-.246	.806	-.264
172.00	.000	-.061	.120	-.507	.612	-.295
174.00	.000	-.029	.120	-.246	.806	-.264
175.00	.000	-.029	.120	-.246	.806	-.264
180.00	.000	-.029	.120	-.246	.806	-.264