Hybrid Approach to Word Sense Disambiguation for Hadiyyisa Language Using Supervised Machine Learning Models

ABRAHAM WOLDE LATISO

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

September 2022

Adama, Ethiopia

Hybrid Approach to Word Sense Disambiguation for Hadiyyisa Language Using Supervised Machine Learning Models

Abraham Wolde Latiso

Advisor: Dr. T.Gopi Krishna

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

September 2022

Adama, Ethiopia

Declaration

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

_____	_____	_____
Name of the student	Signature	Date

Recommendation

I, the advisor of this thesis, hereby certify that I/we have read the revised version of the thesis entitled “Hybrid Approach to Word Sense Disambiguation for Hadiyyisa Language Using Supervised Machine Learning Models” prepared under my/our guidance by Abraham Wolde submitted in partial fulfillment of the requirements for the degree of Master's 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.

_____	_____	_____
Major Advisor	Signature	Date

Approval Page of M.Sc. Thesis

I/we, the advisors of the thesis entitled "**Hybrid Approach to Word Sense Disambiguation for Hadiyyisa Language Using Supervised Machine Learning Models**" and developed by Abraham Wolde hereby certify that the recommendation and suggestions made by the board of examiners are appropriately incorporated into the final version of the thesis.

_____	_____	_____
Major Advisor	Signature	Date

We, the undersigned, members of the Board of Examiners of the thesis by Abraham Wolde have read and evaluated the thesis entitled "**Hybrid Approach to Word Sense Disambiguation for Hadiyyisa Language Using Supervised Machine Learning Models**" and examined the candidate during the 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

ACKNOWLEDGMENT

Primarily, I thank God for donating good health to me, which enabled me to excel in all of my academic endeavors. Then, from the beginning to the end, I would want to convey my deep gratitude to Dr. T.Gopi Krishna, whose knowledge was crucial in providing the proper direction, useful suggestions, and participation.

My heartfelt gratitude also goes out to Hadiyyisa language specialist, Mr. Markos Hegano, and the Hadiya zone education team for their dedication to annotating and providing complete unstructured data for our study. I would like to thank Mr. G/medin pawlos, the head of the Hadiya zone tourism office, for annotating the ambiguous words with their meanings and for his assistance. In addition, I would want to express my gratitude to my beloved and missed mother, whom I lost to W/Menifas Gimiso, for wishing me a happy future even in her darkest days. My mum declared, "The future is bright." This advice is extremely important to me.

Next, I would want to thank my family, especially my aunt, Ms. Beyenechi Gimiso, for their constant advice throughout my life. May God continue to protect and bless them all. Thank you so much!

Table of Contents Pages

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS AND ACRONYMS

AI	Artificial Intelligence
Bi GRU	Bi-Directional Gated Recurrent Unit
BiLSTM	Bi-Directional Long Short-Term Memory
BOW	Bag of Word
LBFGS	Limited-memory Broyden-Fletcher-Goldfarb-Shanno
CBOW	Continuous Bag of Word
DT	Decision Tree
GCN	Graph Convolutional Network
GB	Gradient Boosting
HTV	Hadiyyisa Television media
MRD	Machine Readable Dictionary
MT	Machine Translation
NER	Name Entity Recognition
NLP	Natural Language Processing
NN	Neural Network
NB	Naïve Bayes
RF	Random Forest
TF-IDF	Term Frequency- Inverse Document Frequency
SG	Skip-Gram
SVM	Support Vector Machine
WSD	Word Sense Disambiguation

ABSTRACT

Word sense disambiguation is a core problem in many tasks related to natural language processing. The absence of automatic word sense disambiguation for Hadiyyisa language can be a challenge for the development of natural language processing applications such as text retrieval system, text processing, and automatic sentence parser. In addition to that, some challenges hamper the effectiveness of WSD applications for the Hadiyyisa language, such as ambiguity, the stop word identification problem, and the limitation of digital datasets. As a solution to this, a word sense disambiguation model has been developed for Hadiyyisa language to address the problem of lexical ambiguity. To conduct the research, a lexical sample tagged datasets for two senses of an ambiguous word are considered and used to evolve an optimized hybrid model that correctly disambiguates the sense of the given word considering the context in which it occurs. A new model has been developed by using supervised machine learning models and proposed model have been compared with several existing models. The performance is evaluated with their accuracy, precision, recall, and f1-score including the confusion matrix considered for accuracy comparison. The performance evaluation results revealed that the proposed Hybrid (SVM-NB-NN) model gives better performance for classifying the correct senses of classes than existing models.

Key Words: - Natural Language Processing, Supervised ML, TF-IDF, and Word Sense Disambiguation

CHAPTER ONE

1. INTRODUCTION

1.1. Background of the Study

In natural language processing, word sense disambiguation is defined as the task of finding the sense of a word in a context. It is the process of determining the most appropriate interpretation of an ambiguous word (Popov, 2018)(Zhang et al., 2021)(Chaplot and Salakhutdinov, 2018)(Escudero et al., 2000). Due to the complexity and ambiguity of natural languages, WSD is crucial for NLP. For instance, lexical ambiguity means a word that could be used as a verb, noun, or adjective(Bade, 2021). Lexical ambiguity exists in the occurrence of two or more possible meanings of the sentence within a single word. Open-class words frequently have many meanings because of homonymy or polysemy, which refers to the existence of distinct but related meanings for the same word (the phenomenon of semantically unrelated concepts being expressed with the same word)(Popov, 2018) (Li and Suzuki, 2020)(Bhadane et al., 2021). For developers of natural language processing systems, word sense ambiguity is an unbreakable bad-behaved. A word with an ambiguous sense could confuse users (Chaplot and Salakhutdinov, 2018)(Nithyanandan and C, 2019)(Li and Suzuki, 2020). In addition, natural language is ambiguous, so many words can be interpreted in multiple ways depending on the context in which they occur. To solve the problem of ambiguity, WSD is introduced (Y. Tesema, 2016)(Mukti Desai, 2013)(Bhadane et al., 2021)(Chen, 2019).

Many researchers have conducted research on word sense disambiguation in different languages using different types of methodologies. One of the better resources to disambiguate words is WordNet (Y. Tesema, 2016) (Hassen, 2015)(Chen, 2019) (Nithyanandan and C, 2019). This study focuses on the problem of WSD using supervised machine learning methods. Therefore, to improve WSD, they use labeled data or sense-annotated datasets based on supervised machine learning methods. Supervised machine learning methods are based on the assumption that context can provide enough evidence on its own to disambiguate words. Probably every machine learning algorithm has been applied to WSD, including associated techniques such as parameter optimization and ensemble learning. Therefore, support vector machines are the most successful approaches to date, probably because they can manage the high-dimensionality of the feature space. However, supervised

machine learning methods are subject to a new knowledge acquisition bottleneck since they rely on substantial amounts of manually sense-tagged corpora for training, which are laborious and expensive to create (Han and Shirai, 2021) (Bhadane et al., 2021).

Hadiyya language is not endangered from eighty (80) Nations Nationality of Ethiopia. Thus, currently Hadiyya nations geographically located in the Ethiopian Southern Nations Nationalities and Peoples Region (SNNPR) Hadiyya Zone. Hadiyya people have their own language, historical origin, development and located in different regions of Ethiopia. The central town Wachamo/Hossana/ is located in the north center of the Hadiyya zone at a distance of 232 km south of Addis Ababa and 160 km west of Hawassa town. The Hadiyya language is used by the native people which is more than one million and neighboring people also communicate as their second language. The Hadiyya zone neighboring people are different nationalities surrounding this zone, (to the North by Silti and Gurage, to the south by Wolayitta, to the southeast by Kambaata, by Tambaaro to the south west and in the west by Omo river which separates it from Oromia region and the Yem Special Woreda) (Tadesse, 2015). In addition, the Hadiyya language speakers /users / are also available in the Benishangul Gumuz Regional state at Pawe Woreda and Ormiya regional state around illubabor. (Woldeyohanes, 2016)(Dereje, 2017). Finally, this study has focused on the Hadiyyisa language and it addresses the lexical sample words in the Hadiyyisa language. For example, let us see the ambiguous word "diinate" in the following sentences. "Lambaami ulluminse diinate mine agisukko." In this sentence, the meaning of Diinate is "Hoora (animals)". "Lambaami banki diinate fisukko." In this sentence, the meaning of Diinate is "biira" (money). This ambiguous word can be understood with the following sentence to interpret the correct senses by considering the clue words or context words. The main purpose of WSD is to learn how to give the correct sense of meaning according to the context of words.

1.2. The motivation of the Study

The main motivation of the study is to investigate the supervised ML model and hybrid model to WSD for Hadiyyisa language. However, due to a large number of polysemous words in the language, the automatic processing of Hadiyyisa text is facing difficulties. This motivates me to design and build a WSD model for Hadiyyisa language. Another motivation

of this study is used for Hadiyyisa language speakers like students, teachers, and other personnel in lessons to disambiguate terms simply or understand them easily.

1.3. Statement of the Problem

In natural language processing, languages have several ambiguous words. Many challenges tackle the effectiveness of NLP applications in the Hadiyyisa language. These challenges are ambiguity, stop word identification problems, and the limitation of digital datasets. The lack of automatic WSD for any language can be a challenge for the development of natural language processing applications such as text retrieval system, text summarization, speech processing, text processing, POS, automatic sentence parser, spelling checker, etc (Alemu and Fante, 2021)(Tesema, 2016)(Sreenivasan and Vidya, 2016) (Hassen, 2015)(Rahman and Borah, 2021) (Chen, 2019). The lack of WSD would affect the above-mentioned natural language processing applications.

This study is supported by the existing research that has been done on the Hadiyyisa language. For example, users who look for Hadiyyisa documents with Hadiyyisa queries may not get the system that searches for relevant a document that satisfies their information needs. From this point of view, WSD plays an essential role in having a text retrieval system that works for the Hadiyyisa Language. Ambiguity has to be resolved in some queries. For instance, if the given query contains the ambiguous word "Diinate", should the system return documents about **biira** or **hoora**? Current information retrieval systems, such as Web search engines, rely on the user typing enough contexts in the query to only retrieve documents relevant to the intended sense. All the ambiguous words are disambiguated by the intended senses co-occurring in the same documents. The removal of stop words from indexing and queries results in the effectiveness of retrieval by reducing storage requirements and increasing the matching of a query with index terms of a document (Bade, 2021). The implementation of this work helps us, users of the Hadiyyisa Language, to understand information needs without much difficulty, and it enhances the development of Hadiyyisa to grow with current information technology support (Woldeyohanes, 2016). WSD is useful for automatic sentence parsing that improves scientifically well formulated to develop an efficient parser for the Hadiyyisa language. Because WSD is effective for automatic sentence parsers, a lexical level organizes and analyzes a sentence with a correct sense of meaning (Dereje, 2017).

Finally, WSD has an important role in the existing NLP applications in the Hadiyyisa language. WSD helps to solve the ambiguity of words for each NLP operation based on their tasks. As a result, this research has supported the acceleration of existing Hadiyyisa language research. So, they argue that integrating word sense disambiguation of Hadiyyisa language with the above-mentioned existing NLP applications increases the performance of the NLP applications. Therefore, the lack of WSD in the Hadiyyisa Natural Language Processing system has limited the current work and future efforts to learn computers to understand Hadiyyisa text. Word sense disambiguation is still a working research area in every language using a different methodology. This research focuses on the problem of WSD using supervised machine learning methods. So, currently, no WSD research has been done in the Hadiyyisa language. This motivates me to study the research on WSD. Therefore, WSD is also required to facilitate the existing NLP research for the Hadiyyisa language, and this study mainly uses supervised machine learning setups and a hybrid model to predict the result. By using a voting classifier two or more different supervised machine learning models have been combined.

1.4. Research Questions

Q.1.Which algorithm provides better performance for Hadiyyisa language WSD from the supervised machine learning models?

Q.2.Which feature extraction technique for Hadiyyisa language should be applied?

Q.3.How can the overall performance of WSD for Hadiyyisa language be enhanced?

1.5. Objective of the Study

1.5.1. General Objective

The general objective of this study is to build a Hybrid of supervised ML Models word sense disambiguation for Hadiyyisa language.

1.5.2. Specific Objectives

To meet the above general objective, the study must address the following specific objectives:

- To examine WSD-related works in other languages.
- To gather ambiguous words from a variety of data sources.
- To prepare a manually annotated corpus from a variety of data sources for selected ambiguous words of the language.

- To analyze the data preprocessing phase for the Hadiyyisa text.
- To build a WSD model for the Hadiyyisa language.
- To evaluate and test the performance of the WSD model.

1.6. Significance of the Study

The significance of this study is that different stakeholders, such as Hadiyyisa speakers and other new Hadiyyisa language learners, are used to understanding the proper word meanings. Another benefit of the study is correctly identifying and knowing the meaning of polysemous words in varied contexts. The findings of this study have benefits for the NLP applications such as sentence parser, text retrieval, POS, spelling checker, and lexicography. Besides, to develop a WSD technique that can handle some of the issues with better accuracy and performance. The significance of WSD is to identify the suitable senses of an ambiguous word in a sentence. The target beneficiaries of this study are the Hadiyyisa language speakers, the Wachemo University Hadiyyisa linguistic department lecturers and students, and the Hossana Teachers Training College Hadiyyisa departments. Finally, the overall beneficiaries of this study are the Hadiya zone education sectors.

1.7. Scope and Limitations of the Study

1.7.1. Scope

This study has been limited to the Hadiyyisa language, not including other languages. However, this study has been bound to work at the lexical-semantic level and on the occurrence of related words in the given sentence, phrase, or synonym words. There are two popular variations of training sense-Val for WSD: lexical sampling and all-word. In addition, it is also limited to the lexical sampling tasks, which are focused on a limited number of words sampled in some way from a lexicon (therefore, usually target words appear alone in the training and testing sentences), but do not include all-words tasks. In addition, this study has been limited to the lexical ambiguity type but does not include other types of ambiguity. In this study, a feature extraction technique (TF-IDF) has been used to convert raw text into machine-readable form. Here also apply, evaluate, and compare a range of supervised machine learning methods to Hadiyyisa lexical sampling, including Naïve Bayes, support vector machine, neural network, decision tree, random forest, gradient boosting, and ensemble learning. Therefore, the evaluation of WSD for the Hadiyyisa language has been verified with these models of SVM and a hybrid model. However, in this work, three

supervised machine learning algorithms were combined by using a **voting classifier** technique to provide a **hybrid model** which consists of SGD classifier, mutinomialNB classifier, and MLP classifier.

1.7.2. Limitation

Due to time constraints and the unavailability of the annotated dataset, only ten ambiguous words have been selected with their two meanings. The main reason was a lack of meaningful annotated data and a lack of NLP resources used for WSD, such as Word Net and MRD for the Hadiyyisa language, which forced me to limit the data collection to **2,872** sentences. Moreover, it deals only with textual data formats without any kind of grammar and spelling correction.

1.8. Methodology of the Study

As a result, the overall methodology is investigational research. To do investigational research, the researcher uses the following instruments, techniques, and procedures as follows:

Literature Review:-Various literature works have been reviewed to gain a better understanding of the study and to gain a thorough understanding of the various methodologies that are required for WSD. Various related literature reviews have been studied in the following areas:

- ✓ For both local and foreign languages, word sense disambiguation research should be reviewed.
- ✓ Examine the methodology, research gap, results, advantages, and disadvantages of scientific journals.
- ✓ The Hadiyyisa writing method, punctuation marks, and grammatical structure of the language have been reviewed.
- ✓ Various documents about Hadiyyisa have been surveyed.

Data Source and Dataset Preparation

Preparing the dataset is the most important process in achieving the study goal. Raw text data collection is one of the resources required for natural language processing research. The researcher gathered example texts or sentences from various data sources for this study, including online and offline data sources Institution of Hadiya Radio; Hadiya Television Media; Hadiya Zone Education Bureau; and Hadiya Zone Culture and Tourism Bureau. The reasons for collecting data from these data sources are: First, these ten ambiguous words are

frequently found in that data source and most of the time people are used for communication purposes. The second reason is many sentences that contain ambiguous words have been found most of the time probably from the above-mentioned data sources. Then, the training of WSD for the Hadiyyisa language has been prepared in a lexical sample format. Because there are not enough resources for the Hadiyyisa language like WordNet, Sense Val/sem-core, and the dataset is not found publicly. For that reason, for the Hadiyyisa language, there are no well-organized data resources for WSD compared to other resource-developed languages. Therefore, this study has been working on the lexical sample training of WSD for the Hadiyyisa language.

Methodology

This study is based on the basic variant of the WSD task for the Hadiyyisa language considering the lexical sample task. In this study, the experimental setup to investigate the model using the supervised machine learning setup. The supervised machine learning approach uses a training dataset or tagged dataset with their sense which means a manually annotated dataset and It is used to train a classifier.

Model Selection

This study has been established on supervised machine learning models and hybrid models. From the supervised machine learning setup, such as Naïve Bayes, Support Vector Machine, and Neural Network. From these mentioned supervised machine learning setups, after verifying the result of each model. Then SVM model has been found an encouraging result. Moreover, the ensemble of supervised machine learning algorithms uses classification algorithms such as Decision Tree, Random Forest, and Gradient Boosting. From the above-mentioned Ensemble Supervised machine learning setup, after verifying the result of each model, the Random Forest model has been providing promising results. Finally, in the hybrid model, the integration of different supervised machine learning models of NB-SVM-NN, NN-NB, NB-SVM, and SVM-NN was verified. From these mentioned hybrid models, after verifying the result of each model's **NB-SVM-NN** model have been provide the best performance results.

Evaluation Techniques

To validate a model, using the Monto Carol (Shuffle Split Cross-Validation Techniques), i.e., the 5-fold shuffle split and 10-fold shuffle split methods have been used to validate the performance of the model. After the experimental evaluation result, the 10-fold shuffle split method provided more validation accuracy than the 5-fold shuffle split method. Finally, the

overall performance of the model was evaluated by the following evaluation metrics: accuracy, precision, recall, and F1-score

1.9. Thesis Organization

This section describes the organization of the rest of the thesis:

Chapter two discusses the literature review, past efforts on word sense disambiguation, the nature, and structure of the Hadiyyisa language, the Hadiyyisa writing system, the morphological structure of the Hadiyyisa language, and linguistic ambiguities. The basic systematic and comparative literature review for word sense disambiguation in foreign and local languages has been discussed.

Chapter three discusses the research methodology utilized to design the proposed work flow. It includes a data collection mechanism, corpus preparation, data preprocessing, feature extraction technique, and the principles of research materials, techniques, and methodology to gain a thorough understanding of the research methodology used in past research as well as the research methodology recommended for our thesis study.

Chapter four discusses the design, WSD system architecture, and basic components of the proposed WSD model.

Chapter five discusses the implementation of WSD for the Hadiyyisa language. And it discusses WSD implementation and the basic necessary steps to develop a model, from starting data loading up to analyzing results.

Chapter six discusses the study results and discussion, and the experimental findings are mainly presented.

Chapter seven discusses conclusions, recommendations, and future research directions.

CHAPTER TWO

2. LITERATURE REVIEW

In this chapter, a lot of work carried out on word sense disambiguation should be reviewed to identify the gaps and findings of the research and to have a deep understanding of the concepts and methods of WSD in foreign and local languages. For example, to learn about the fundamental approaches in WSD and to review the most recent research papers on foreign and local languages. In addition, different books, journals, articles, videos, and notes have been reviewed to understand the general concepts of word sense disambiguation. A literature review is also used to identify research gaps and objectives, as well as how to generate WSD datasets for other languages that have been important to this research study.

2.1. Overview of Natural Language Processing and Word Sense Disambiguation

In natural language processing, WSD means the process of determining the most appropriate interpretation of an ambiguous word (Popov, 2018). As a result, it is one of AI's most difficult challenges (Loureiro et al., 2021) (S. Gupta et al., 2017). The basic task of the WSD is to identify the suitable sense of an ambiguous word in a sentence (Vial et al., 2020) (Loureiro et al., 2021). Word sense disambiguation is an important method of NLP by which the meaning of a word is determined, which is used in a particular context. NLP systems often face the challenge of properly identifying words, and determining the specific usage of a word in a particular sentence has many applications. To disambiguate a word, they needed two basic things: (1) a dictionary-based approach, and (2) a corpus-based approach. A dictionary-based approach means a collection of senses with their ambiguous words, which means semantic relations of polysemous word dictionaries such as WordNet, machine-readable dictionaries, and so on (Vial et al., 2020)(Escudero et al., 2000). A corpus-based approach means a collection of the real word context of the ambiguous words; supervised and unsupervised methods rely on corpus evidence (Escudero et al., 2000) (Taghipour and Ng, 2015)(Bhadane et al., 2021) (Karwa and Chandak, 2015).

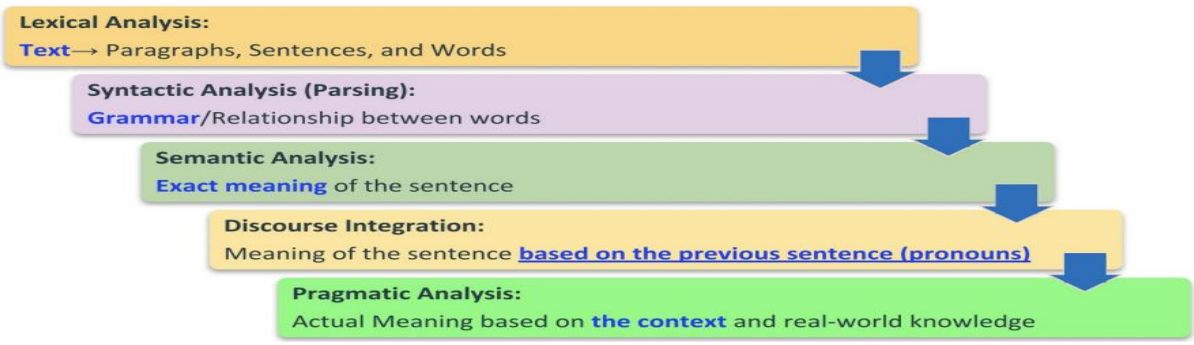


Figure 2. 1 Stage in Natural Language Processing[1]

Common preprocessing in NLP [2]

Noise removal includes removing whitespaces, HTML tags, accented characters, special characters, and stop words (not always).

Normalization: - uppercase characters convert lowercase characters.

Vocabulary: - get unique tokens from the corpus with their frequencies.

Vectorization: - It uses word embedding techniques or a bag of words.

Figure 2. 2 Common Preprocessing in Natural Language Processing

2.2. History of Word Sense Disambiguation

WSD was first formulated as a distinct computational task during the early days of machine translation in the 1940s, making it one of the oldest problems in computational linguistics. The issue was first presented in a computational setting by Warren Weaver in his renowned 1949 brief on translation. Early researchers had a clear understanding of the importance and complexity of WSD (Han and Shirai, 2021) (Kilgarriff, 1998). Bar-Hillel (1960) employed the above scenario to argue that WSD could not be solved by an “electronic computer" because of the need in general to model all world knowledge (Kilgarriff, 1998). Since WSD systems were mostly rule-based and hand-coded in the 1970s, they were vulnerable to a knowledge

[1] http:www.tutorialspoint.com/artificial intelligence/ artificial intelligence natural language processing.htm

acquisition bottleneck. WSD systems were developed as a subtask of semantic interpretation systems within the field of artificial intelligence (Patil et al., 2020)(Rahman and Borah, 2021)(Chen, 2019). When extensive lexical resources, such as the Oxford Advanced Learner's Dictionary of Current English (OALD), were made available in the 1980s, hand-coding was replaced by the knowledge that was automatically pulled from these resources, but disambiguation was still knowledge-based or dictionary-based. In the 1990s, the statistical revolution swept through computational linguistics, and WSD was transformed into a paradigm problem that could be solved using supervised machine learning methods. The 2000s saw supervised techniques reach a plateau in accuracy, and so attention has shifted to coarser-grained senses, domain adaptation, semi-supervised and unsupervised corpus-based systems, combinations of different methods, and the return of knowledge-based systems via graph-based methods. However, supervised systems continue to have the best performance (Patil et al., 2020) (Sharma and Niranjan, 2015) (Edmonds 2005) (Patil et al., 2020).

2.3. Resources for Word Sense Disambiguation

The selection of word senses means it also needs the utilization of external information sources and the indication of context (Navigli, 2009). In general, WSD resources have been working on a knowledge-intensive task that uses two types of data resources:

Structured resources

Sense inventories reference computational lexicons that list probable meanings for words (Navigli, 2009). The **Thesaurus** provides information on word relationships such as synonymy (e.g., diinate is a synonym of Biira, miin hoora, and amaxxa), and antonym (e.g., wo'o is an antonym of Qoxara). For example, **MRD** is a popular source of knowledge for natural language processing; among these, they cite the Collins English Dictionary, the Oxford Advanced Learner's Dictionary of Current English, and the Oxford Dictionary of English (Soanes and Stevenson,2003) (Navigli, 2009)(Chaplot and Salakhutdinov, 2018). The most popular sense inventory resources that are used for foreign languages such as English are:

- ✓ Princeton WordNet has a huge, carefully compiled lexicographic database of English that serves as the de facto standard inventory for WSD and other WSD-related projects. It has also been organized as a graph, with nodes representing synsets, or collections of contextual synonyms. Each synonym in a synset

represents one of the word's meanings (Vial et al., 2020)(Chaplot and Salakhutdinov, 2018)(Bevilacqua et al., 2021).

- ✓ BabelNet has a multilingual dictionary with lexicographic and encyclopedic coverage produced by semi-automatically mapping numerous resources, including WordNet, multilingual versions of WordNet, and Wikipedia and among others. Babel Net is organized as a semantic network that contains nodes that represent multilingual synsets (groups of synonyms lexicalized in many languages) and edges represent meaningful relationships between them (Loureiro et al., 2021)(Bevilacqua et al., 2021).

Unstructured resources

Annotated corpora: a word is labeled with one or more possible meanings selected from the inventory, and it is also used for training and testing WSD systems (Bevilacqua et al., 2021).

- ✓ A corpus means a collection of texts that we use for language research. Corpora can be raw or sense-annotated text (i.e., unlabeled) (Navigli, 2009)(Vial et al., 2020).
- ✓ The Brown Corpus (Kucera and Francis, 1967) is a million-word balanced collection of texts published in the United States in 1961. The Brown Corpus (Kucera and Francis, 1967) is a million-word balanced collection of texts published in the United States in 1961.
- ✓ The Gigaword Corpus (Graff, 2003), contains 2 billion words of newspaper text.
- ✓ The American National Corpus (Ide and Suderman, 2006), comprises 22 million words of written and spoken American English (Navigli, 2009).
- ✓ Sense-Annotated Corpora: With 352 texts and about 234,000 sense annotations, SemCor is the largest and most commonly used sense-tagged corpus (Navigli, 2009)(Vial et al., 2020).

2.4. Challenges in Word sense Disambiguation

Why WSD is hard? WSD faces many challenges to study:

- ✓ The most common problem is the difference between various dictionaries or text corpus. Different dictionaries have different meanings for words, which makes the sense of the words be perceived as different. A lot of textual information is out there and often it is not possible to process everything properly (Zeynep, 2010)(Muruts, 2018)(Nancy Ide, 2004)(Sharma and Niranjan, 2015).

- ✓ Different applications need different algorithms and that is often a challenge for WSD (Zeynep, 2010)(Xue-Ren, Shao-he Lv, 2017)(Nancy Ide, 2004)(Sharma and Niranjan, 2015).
- ✓ A problem also that arises is that words cannot be divided into discrete meanings. Words often have related meanings, and this causes many problems (Zeynep, 2010)(Nancy Ide, 2004)(Sharma and Niranjan, 2015).

2.5. Types of Feature-Based Word Sense Disambiguation

Lexical features: words that are near the target ambiguous words. It comprises unigrams and bigrams with window sizes of two, three, and four, as well as collocations and co-occurrence.

I.e. $unigram(w_t) = U^{m}_{x=1}\{w_1, w_2, ...w_k \in S\}$ (2.1)

In the context of an instance, the Bigrams features have two-word sequences. For instance, bigrams encompass the target uncertain word as well as all adjacent words.

I.e. $Bigrams(w_t) = U^{m}_{x=2}\{w_1, w_2, w_k \in S\}$ (2.2)

The words in an orderly succession make up the collocation features.

I.e. $coll(w_t) = U^{m}_{x=1}\{U^{m}_{x=2}(w_{-2}, w_t, w_{+2}) \in S\}$ (2. 3)

It also contains the ambiguous target word wt, as well as one word preceding (w1) and one word after (w+1). The characteristics of the target term include its distinctive collocations. Co-occurrence refers to the recurrence of indicative words in the context of an ambiguous word. In the middle, the number of words varies (Navigli, 2009) (Singh and Kumar, 2019).

$$Coccur(w_{i,} w_t) = \sum_{sk} w_i \quad (2.4)$$

Syntactic features: based on the language grammar rules, the shallow parser captures the precise syntactic relations between the target word and its nearby words, and the shallow parser additionally assigns POS tags to the context words of the training and testing data sets (Navigli, 2009)(Singh and Kumar, 2019).

Word embedding features: It captures the semantic relationships between words and is the most successful distributional semantics technique and is effectively integrated with the supervised WSD system for the English language (Wahle et al., 2021)(Eshetu et al., 2020). The details of these procedures are as follows:

1. **Word2vec** is a three-layer NN-based technique for generating word embedding. Continuous Bag of Word (CBOW) and Skip-Gram are two different versions of

Word2Vec and SG. By looking at its context words, the CBOW model creates a single target word. Within a symmetric window, the skip-gram model determines the target words (Navigli, 2009)(Singh and Kumar, 2019)(Eshetu et al., 2020)(Melamud et al., 2016).

2. **Fast text:** The word2vec represents a single word as a single vector, ignoring the words' underlying structure (Singh and Kumar, 2019)(Eshetu et al., 2020).

2.6. Applications of Word Sense Disambiguation

NLP is an application of artificial intelligence and offers the facility to companies that need to analyze their reliable business data (Bade, 2021). A successful natural language processing application, therefore, must also choose the correct meaning of a word from that variety of possibilities(Sharma and Niranjan, 2015). Some of the applications of NLP are as follows.

A. **Information Retrieval**

Information retrieval is a significant technological advancement in human history. It is the primary technological component of search engines and is used often by many online users. IR uses both structured and unstructured data to search. Searching and indexing are the two primary subsystems of the entire IR system (Woldeyohanes, 2016).

The investigations demonstrate that even in a specialized database, there remains an important amount of ambiguity. Word senses significantly distinguish between relevant papers and those that are not, but several factors contribute to determining whether disambiguation will improve performance (P. Gupta et al., 2013). There are three reasons why WSD should be utilized for IR: First, if the queries are short, the context for context-based query phrase disambiguation is fixed, making WSD troublesome. Second, for terms with many meanings, the most common sense often dominates the frequency distribution of the text collection in question; in these circumstances, correct disambiguation is most likely to be obtained by using the word itself. Third, document retrieval models are prone to implicit disambiguation of multi-word queries, which improves bag-of-words IR performance in the absence of explicit word senses, particularly for longer queries (Patil et al., 2020) (Sharma and Niranjan, 2015)(Tesfaye, 2017).

B. Machine Translation

In machine learning, WSD should be used to determine the correct Interlingua or "meaning" representation for a concept in the source language. Because of these factors, WSD for MT

systems is unlikely to succeed. To begin with, since ambiguity is only one of several sorts of ambiguity that MT had to deal with; therefore, WSD cannot be investigated separately in this circumstance. Second, standardized MT evaluations at the community level are relatively new phenomena. For a variety of reasons, evaluating the effectiveness of explicit WSD in ordinary MT systems is difficult. To begin with, sense ambiguity is just one type of ambiguity that MT systems must contend with, which may explain why WSD is rarely considered a discrete component. Second, standardized, community-wide MT evaluations are a relatively new phenomenon. Explicit WSD does not appear to have played a visible role in any of the early 1990s systems (White and O'Connell, 1994), and statistical MT systems have been the majority of present comparative evaluation participants (Patil et al., 2020) (Sharma and Niranjan, 2015)(Tesfaye, 2017).

C. Information extraction (IE) and Text mining

Many information extraction and text mining situations, such as bioinformatics research, named entity recognition systems, and co-reference resolution, rely on WSD research to discriminate between specific occurrences of concepts (Patil et al., 2020)(Sharma and Niranjan, 2015)(Tesfaye, 2017).

2.7. Approaches Used for Word Sense Disambiguation

There are three basic methodologies employed in WSD's activities: Knowledge-based, corpus-based, and hybrid techniques.

2.7.1. Knowledge-Based Approach

The knowledge-based method decodes words by comparing the context to a known source of information. Since any electronic information source, such as Word Net, can be utilized to integrate the dictionary and structured semantic network properties (Kilgarriff, 1998)(Navigli, 2009) (Assemu, 2011)(Chaplot and Salakhutdinov, 2018) (Dawn, 2020). It also needs an MRD. To find the overlap between the sense Bag (features of a different sense of an ambiguous word) and the context Bag (features of the words in their perspective), use external lexical resources such as dictionaries, ontology, collocations, and thesauri (Gottschalk et al., 2002)(Navigli, 2009)(Li and Suzuki, 2020)(Bhadane et al., 2021) (Bala, 2013)(Pal and Saha, 2015).

2.7.2. Corpus-Based Approach

One of the challenging areas of WSD is the annotated corpus or context-based repository. When they learn from previously sense-annotated data, they usually require a large amount of human intervention to annotate the training data. It represents linguistic information in the form of feature vectors. It is not essential to annotate these feature sets for all contexts of each sentence (Solomon, 2010)(Loureiro et al., 2021)(Wahle et al., 2021)(Assemu, 2011) (Bhadane et al., 2021)(Alemu and Fante, 2021)(Rahman and Borah, 2021).

Supervised Machine Learning Methods for Word Sense Disambiguation

A classification challenge using machine learning algorithms from manually annotated data is referred to as a supervised approach. It also includes a training phase as well as a testing phase (Navigli, 2009) (Solomon, 2010)(Dawn, 2020)(Taghipour and Ng, 2015)(Abid et al., 2018)(Iacobacci et al., 2016). They require a significant level of human intervention to annotate the training data. Training sets are examples of ambiguous phrases that are used to teach the classifier during the learning phase (Vial et al., 2020) (Solomon, 2010) (Taghipour and Ng, 2015) (Alemu and Fante, 2021). As evidenced by public evaluation activities, the highest performing WSD systems are currently those based on supervised learning (Eneko Agirre, 2010). The WSD uses a manually labeled corpus for training and testing the contextual sentences in the supervised method (Papandrea et al., 2017) (Melamud et al., 2016) (Alemu and Fante, 2021) (Abid et al., 2018). It's difficult to annotate a large volume of data and collect enough contextual data for every possible meaning of a word in natural languages (Bala, 2013)(Tadesse, 2021)(Abid et al.,2018)(Taghipour and Ng, 2015)(Iacobacci et al., 2016). Some of the supervised machines learning (ML) techniques are discussed below

1. **Naïve Bayes**

It is a probabilistic classifier based on Bayes' theorem application. It's a supervised machine-learning technique based on Bayes' Theorem, which implies that features are statistically independent (Abid et al., 2018)(Navigli, 2009).

- Maximum Posteriori Hypothesis

$$P(Ci|X) = \frac{P(X|Ci)P(Ci)}{P(X)} \qquad (2.5)$$

Maximum $P(Ci|X) = P(X|Ci)P(Ci)$ as P(X) is constant; where

$$\text{Posterior} = (\text{likelihood} * \text{prior}) / \text{evidence}$$

- ❖ To determine the meaning of the word w, the sense with the highest probability has been picked from each sense Si, and attributes associated with the context of w have been chosen.
- ❖ Works with: Nominal values, numerical.

For a text and document categorization, multinomial naive Bayes has been utilized, which means discrete values or counts. The Gaussian naive Bayes classifier, on the other hand, is used to categorize continuous data, whereas the Bernoulli naive Bayes is used to classify Boolean data that is either True or False (Navigli, 2009) (Dawn, 2020)(Tadesse, 2021).

2. Decision Tree

A supervised non-parametric learning method that is used for classification and regression is known as the decision tree. It is a series of if-then-else rules for predicting a target variable based on data attributes. The training dataset is used to create these if-then-else rules, to satisfy as many training data instances as feasible (Navigli, 2009)(Tadesse, 2021) (Thankappan et al., 2016). An advantage of decision trees is that they can model any type of function for classification or regression, which other techniques cannot (Thankappan et al., 2016). The decision tree provides principles for disambiguating ambiguous words.

$$Entropy(S) = \sum -p(I) * \log_2 p(I) \tag{2.6}$$

The attribute having maximum Information gain has been used to split the dataset S on that particular iteration. It is written mathematically represented as -

$$Gain(S, A) = Entropy(S) - \sum \left[(S|A) * Entropy(S|A) \right] \tag{2.7}$$

DT analysis is a predictive modeling approach that has been used in a variety of situations. It should be built using an algorithm that divides the dataset in different ways depending on the conditions. Both classification and regression tasks are performed with it.

3. **Support Vector Machine (SVM)**

It is a classification algorithm for both linear and non-linear data. It translates the original data into a higher dimension, from which it can create a hyperplane for class separation using critical training tuples known as support vectors. The SVM looks for the hyperplane with the greatest margin, which is the one with the greatest distance between training tuples. The association margin provides the most class distinction (Dawn, 2020). The key goal is to increase this distance while reducing classification errors. In the SVM, each sense of a word is treated as a separate class, with the remaining senses being treated as members of the same class (Tadesse, 2021).

4. Neural Network

It is designed to organize information, discern patterns, and interpret sensory information. Despite their numerous benefits, neural networks need a significant amount of computing power (Dawn, 2020)(Vial et al., 2020). When there are thousands of observations, fitting a neural network might get difficult. It is a "black-box" algorithm since deciphering the rationale behind its predictions can be difficult.

5. **Random Forest Classifier (RFC)**

It creates a set of decision trees from a randomly selected subset of the training data. It determines the test object's final class by combining votes from several decision trees. The original parameters had been determined by rigorous separate divide and conquer constraints. It is used to solve both classification and regression problems (Tadesse, 2021).

6. Ensemble Methods

When different classifiers are available, integrate them to increase the overall accuracy of the disambiguation process (Navigli, 2009). Alternatively, characteristics should be selected to produce noticeably diverse, potentially independent, perspectives on the training data (e.g., lexical, grammatical, semantic, etc.). Ensemble methods are gaining popularity because they make up for the shortcomings of standalone supervised procedures. Single classifiers can be combined with different strategies: majority voting, probability mixture, rank-based combination, and AdaBoost. Other ensemble techniques, such as weighted voting and maximum entropy combination (Navigli, 2009).

7. **Bagging**

An ensemble method combines the weights of each weak learner to increase algorithm stability and accuracy. Overfitting and variance in the approach can be avoided by using a combination of weak learner classifiers, especially DT classifiers (Tadesse, 2021).

Unsupervised Machine Learning Approach

The unsupervised technique, in contrast to the supervised approach, relies on clustering training data to uncover groups related to several senses. It derives senses from text using similarity measurements and obtains contextual information directly from untagged raw text (Abid et al., 2018). The difficulty has divided the usages of a word into multiple meanings based on this technique, without respect to any present sense inventory. Automatically acquired data, on the other hand, are frequently noisy and inaccurate (Iacobacci et al., 2016)(Rahman and Borah, 2021) (Gottschalk et al., 2002) (Abid et al., 2018). Word sense induction is a technique used in unsupervised machine learning techniques in WSD to

automatically infer the meaning of ambiguous words from examples of words (Iacobacci et al., 2016)(Dawn, 2020)(Tadesse, 2021). Type-based and token-based approaches are two strategies for splitting. The type-based approach of disambiguating a target word by grouping or clustering sentences and the token-based method of disambiguating a target word's context by grouping or clustering sentences (Han and Shirai, 2021)(Alemu and Fante, 2021)(Pal and Saha, 2015). Some unsupervised ML approaches are discussed below.

Context and word Clustering

Context clustering is a methodology that uses clustering techniques to build context vectors, which are clustered into clusters to determine the meaning of a word. This approach employs vector space as a word space, with only words as dimensions. In terms of making sense, word clustering is analogous to context clustering. However, in this case, the technique groups semantically similar words (Tesema, 2016).

Voting classifier

It is a machine learning model that trains on an ensemble of numerous models and **predicts an output (class) based on their highest probability of chosen class as the output**. It simply aggregates the findings of each classifier passed into Voting Classifier and predicts the output class based on the highest majority of voting. By combining two or more models to facilitate the corpus-based method to provide better performance accuracy to handle WSD and vice versa (Assemu, 2011)(Rahman and Borah, 2021) (Bhadane et al., 2021)(Montoyo and Su, 2005)(Demidova et al., 2019).

2.8. Hadiyyisa Language

2.8.1. Introduction to Hadiyyisa Language

The Hadiya people speak Hadiyyisa as a communication language. There are many diverse language groups in the world, but the Hadiyyisa language belongs to the Cushitic language family. It is also one of the under-resourced languages in Ethiopia. It is located in the SNNP. The Hadiya people are located in the Hadiya zone and its boundary with other zones, such as Silta, Kambatta, Gurage, and Woliyita (Dereje, 2017).

2.8.2. Alphabet Development of Roman-Based Hadiyya Orthography

There are 22 consonant morphemes and 5 vowel morphemes created at the start of the Hadiyya orthography's development. The comparable morphemes in the Ethiopic script 29

have been used to create an alphabet chart in the orthography. The initial Hadiyyisa orthography Roman characters are seen in **Annex A** (Dereje, 2017).

2.8.3. The Hadiyya Phonology

It is necessary to understand Hadiyyisa phonology before discussing orthography because it is the foundation for orthography. As a result, the phoneme inventory and syllable structure of the language has been briefly listed in the following sections. Pitch, Diphthongs, Vowels, Consonants, and Gemination. In Hadiyyisa, consonant gemination is a morpheme, meaning that the gemination of consonants changes the meaning, as demonstrated in Table 2.1 Below (Garkebo, 2015).

Table 2. 1 Gemination of Consonant in Hadiyyisa

Status of morpheme	Hadiyyisa		Gloss
Non-geminated	A	[anukko]	'he split'
Geminated		[annukko]	'become matured'
Non-geminated	B	[dasukko]	'delayed'
Geminated		[dassukko]	'to chop off'

In above Table 2.1, geminated consonants are only found in the middle of words. In Hadiyyisa, consonant length influences meaning. Consonants length occurs solely in word middle positions in this language; it does not appear in word start or ending positions (Garkebo, 2015).

Table 2. 2 Gemination of Vowel in Hadiyyisa

Status of morpheme	Hadiyyisa		Gloss
Short vowel	A	[hafa]	'Shadow '
Long vowel		[haafa]	'forgiveness'
Short vowel	B	[gudukko]	'became ready'
Long vowel		[guudukko]	'made burnt'

In above Table 2.2, Geminated vowels are found in the middle and beginning of words. In

Hadiyyisa, vowel length influences meaning. Vowel length occurs in the word middle, beginning, and ending position in this language (Garkebo, 2015).

2.8.4. Hadiyyisa Language Morphology (Sono'o)

Syntax in the Hadiyyisa language: SVO or SOV agreements for sentence formation are used. In the Hadiyyisa language, there is a three-bound morpheme. These are

1. Prefix affix (illaqqi ejjo'o):- means adding the morpheme before one stem word, but it can change the meaning of the stem word.
2. Infix (Bagaagi Somo'o):- means adding the morpheme between the stem words, but it can change the meaning of the stem word.
3. Suffix (lasaqqi som'o):- means adds the morpheme after the stem word, but it can change the meaning of the stem word.

Table 2. 3 Morphemes (Somo'o)

Stem word	Affix Morpheme (Somo'o)	Infix Morpheme (Somo'o)	Suffix Morpheme (Somo'o)
Anna	Iyyanna (iyy-)	-	-
Mure	-	Muramsukko(amse-)	Muramsukko (ukko)
Ite	-	Itamsukko(-amse-)	Itamsukko(-ukko)

2.8.5. Hadiyyisa Language Punctuation Marks

The following are the major punctuation signs in Hadiyyisa orthography as almost similar to English, but only one punctuation mark is different from English, which means bacixxi mare'e (i) is only found in the Hadiyyisa language. The following Table 2.4 shows punctuation marks in the Hadiyyisa language.

Table 2. 4 The Hadiyyisa Punctuation

Symbol	Name of punctuation mark
.	uullishshi mare'e(full stop)
,	giphit mare'e(comma)
;	shiqqeen giphit mare'e(semi-colon)
:	caakkishshi mare'e(colon)
?	xa'michchi mare'e(question mark)
!	maalalaxxi mare'e(exclamation mark)
“”	aggishshi mare'e(quotation mark)
-	ceeqit mare'e(hyphen)
( )	qaapho'o(parenthesis)
< >	matandar aggishshi mare'e(single quotation mark)
—	dufaachchi mare'e/daasha(dash)
i	Becixxi mare'e
‘’	Caqixxi mare'e

(Garkebo, 2015).

2.8.6. Ambiguities in Hadiyyisa Language

In the Hadiyyisa language, there are different types of ambiguity. These are

A) **Lexical Ambiguity**: is a single word's ambiguity. It exists in the presence of two or more possible senses of the sentence within a single word. In terms of syntactic class, a word can be ambiguous. Lexical ambiguity has been handled via lexical category disambiguation, which entails tagging speech pieces. Lexical ambiguity is caused by a variety of circumstances. Now we will look at categorical ambiguity and homonymy. Lexical constituents with the same phonological form but belonging to different word classes cause categorical ambiguity. Homonyms are lexical items that have the same phonological form but different meanings, causing ambiguity (Tesfaye, 2017).

B) **Phonological Ambiguity** arises from the placement of a pause inside a speech structure related to the sound used for the word. With the variation of vowels, the same sounds of words can have different meanings. With the variation of consonants, the same sounds of words can have different meanings (Tesfaye, 2017).

Table 2. 5 Hadiyyisa Ambiguities with Variations of Vowels and Consonants Letter

Hadiyyisa orthography	Variation of vowels and consonants	Gloss
Qotukko	-	'broken partly'
Qootukko	Vowels adding 'o' letter	'(he) gave dowry'
Qottukko	Consonant adding 't' letter	'sat down on ankle'
Golukko	-	'narrowed down'
Goollukko	Vowels adding 'o' letter	'(he) concluded'
Gollukko	Consonant adding 'l' letter	'became greedy'

In the above Table 2.5 Describes it as one of ambiguity in the Hadiyyisa language (Garkebo, 2015)(Godisso, 2017).

C) **Referential ambiguity** exists when you are referring to something using the pronoun and occurs when a word or phrase refers to two or more attributes or items in the context of a sentence (Tesfaye, 2017).

D) **Semantic ambiguity** exists in the presence of two or more possible meanings within the sentence and it refers to the occurrence of a term having several meanings. Polysemy and idiomatic elements are to blame (Tesfaye, 2017).

2.9. Related Work

The related works are important in identifying basic research guidelines or procedures to carry out our research goal and in guiding me to identify research gaps, research methodology, study objectives, and findings. There are several studies on word sense disambiguation, and a few are closely related to our study. This study focused on two aspects for selecting studies for our discussion, namely the language family (local languages such as Hadiyyisa, Tigrigna, Amharic, and Afaan Oromo and other foreign languages such as Telugu, Arabic, Chines, and Indians) and the approaches followed (Corpus-based, Knowledge-based, Hybrid Approach, and Deep Learning Approaches).

The authors of this journal used the term "word sense disambiguation" to describe how they sensed the proper meaning of words depending on given situations using both supervised and unsupervised methodologies. They also employed two approaches: Modified Lesk (ML) and Bag-of-Words (BOW), with "F-Measure" values of 0.65 and 0.57, respectively, whereas the proposed strategy had a value of 0.86 (Heo et al., 2020).

Shotgun WSD methodology has also been utilized to decipher the Telugu language, as well as a combination of unsupervised ML and knowledge-based approaches. They have been used to decipher the Telugu proposed approach, which includes estimating the semantic relationship between the context of the ambiguous word's usage and its sense definitions to extract the ambiguous word's senses from the Telugu corpora. The Telugu language, in addition to using the shotgun sequencing methodology, word embedding using word2vec, and Cosine similarity algorithms, has the highest rate of accuracy and precision. They focus on Telugu, taking Telugu data from the Indo WordNet and creating their sense-annotated corpora. With a shotgun, an accuracy of 80% has been achieved. They identified a research gap by attempting to extend this approach to other Indian regional languages for which WordNet is available (Eluri and Pilli, 2020).

In this research paper, the researcher would also utilize Bidirectional LSTM techniques to disambiguate the phrases, which could improve the experiment's results. Method SE2 and SE3 BLSTM (proposed model) are 66.9 and 73.4, respectively. They proposed a WSD model based on BLSTM that was able to utilize word order effectively and obtain state-of-the-art outcomes on two WSD datasets. The researchers' main contributions to these journals are: - an end–to–end trainable WSD system that achieves state–of–the–art results on two standard datasets without any hand–crafted features, a novel regularization method for word sequences, drop the word, and empirical evidence that highlights the importance of word order for WSD(Mikael and Salomonsson, 2016).

In this research paper, the researcher would also employ the decision tree method from these approaches to disambiguate the phrases, specifically in the REPTree decision tree learner, and it is easy to interpret and comprehend. It requires minimal data preparation and can provide benefits even when there is limited hard data. It can process enormous amounts of data quickly. The model fits for predicting word senses and has been generated using a Weka implementation of the REPTree decision tree learner. The correctness of this work is around 86.00% (Thankappan et al., 2016).

In this research paper, the researcher would focus on WSD's evaluation of an unknown collection of words. Learning models are evaluated in a completely unknown corpus (Senseval 2 English Lexical Sample), overcoming the random baseline, and also implementing all neural network models utilizing libraries to handle embedding: the Numpy

and Genism libraries, as well as the Keras framework. The bidirectional recurrent network based on GRU cells performed best with the mean square error function, followed by the multilayer perceptron with the Poisson error cost function; however, the recurrent-classifier network achieved better precision than the other models (Calvo et al., 2019).

The researcher has developed a hybrid strategy for Afaan Oromo word sense disambiguation that includes unsupervised and rule-based approaches. He also used the Java Net Beans 8.0.2 and the Weka 3.7.9 package for implementation in this investigation. According to this thesis's evaluation metric, the results of hybrid techniques outperform unsupervised and knowledge-based approaches. WSD accuracy for Afaan Oromo is 76.05 % for the unsupervised technique and 94.70% for the mixed approach, respectively (Y. Tesema, 2016).

The researcher has used word sense disambiguation for Ge'ez language using a corpus-based approach and it is used for the Ge'ez data set performing the WSD. In addition, the use of the AD Tree algorithm compared to both classification algorithms and clustering and better results by the Algorithm for Ge'ez language Precision, Recall, F1-score, and accuracy of each parametric 92.1%, 91.3%, 91%, and 91.1%, respectively (Alemu and Fante, 2021).

The researcher has used WSD for the Tigrigna language using semi-supervised approaches. It also uses both clustering and classification algorithms, and five clustering methods such as (EM, Simple K-Means, Farthest First, and Hierarchical Clustered) and five classification methods such as (AD Tree, AdaBoostM1, Bagging, SMO, and Naive Bayes) have been used, with the best results of 93.6636 %, 91.9224 %, 85.35 %, 79.1917 %, and 70.8968 %, respectively (Muruts, 2018).

The researcher for Afaan Oromo WSD, a supervised machine-learning approach has been used, as well as the Nave Bayes theorem, to obtain the prior probability and likelihood ratio of the sense in the given context. Only five ambiguous terms are used, and the available sense-annotated corpora are huge, requiring manual sense labeling (Kebede, 2013).

The researcher for the Afaan Oromo language used unsupervised machine learning and a rule-based approach to disambiguate words. They used tools such as Net beans and Weka 3.6.5 tool to develop the Sense Disambiguation for Afaan Oromo, and there is no annotated corpus for Afaan Oromo (an unsupervised technique that is appropriate when there is a shortage of training data) (W. Tesema et al., 2017).

The researcher for the Amharic language uses an unsupervised learning strategy to convert WSD to Amharic terms, as well as the five ambiguous words, to determine the proper meanings. In addition to testing, the existing implementation employs five clustering algorithms from the Weka 3.6.4 package, including basic k-means, hierarchical agglomerative: Single, Average, and Complete Link, and Expectation Maximization. The best unsupervised Amharic WSD algorithms have an accuracy of 65.1 to 79.4% for Simple k means, 67.9 to 76.9% for Expectation-Maximization, and 54.4 to 71.1 % for Complete Link(Assemu, 2011).

The researcher for Amharic languages developed a knowledge-based WSD approach using Amharic Word Net. In addition, the results of the knowledge-based Amharic WSD have been verified with two experiments. First, he used Amharic Word Net with and without a morphological analyzer; second, he used an optimum window size for Amharic WSD. He used "knowledge-based approaches from these approaches also select Lesk's Algorithm and achieve the result with and without a morphological analyzer an accuracy of 57.5% and 80%, respectively, and in the second experiment he also found a two-word enough window on each side of the Amharic ambiguous words." (Hassen, 2015).

The researcher has used Word Sense Disambiguation for the Afaan Oromo using the Knowledge Base approach and the results can be achieved with and without a morphological analyzer to evaluate the study are 63.95 and 50.75 respectively (Gonfa, 2018).

Table 2. 6 Summary of Related Literature Review

No	Authors and Titles	Methodology	Result	Research Gap
1.	Corpus-Based Word Sense Disambiguation for Ge'ez Language (Alemu and Fante, 2021)	Machine learning techniques (Adaboost, SMO, and AD Tree)	The Precision, Recall, F1-score and accuracy of the achieved result utilizing the AD Tree method are 92.1 %, 91.3 %, 91.0 %, and 91.1 %, respectively.	✓ Words have only two senses. Do more than two senses. ✓ Only modeling WSD to tackle lexical ambiguity, which is at the word level. ✓ Only used corpus-based approaches.
2.	Afaan Oromo Word Sense Disambiguation Using Word Net (Tesfaye, 2017)	Corpus-based and knowledge-based approaches	With a sample of Afaan Oromo WordNet results, the system's accuracy is 92 %.	✓ The researchers only focus on the lexical ambiguity in the Afaan Oromo language.
3.	Hybrid Sense Classification Method for Large-Scale Word Sense Disambiguation (Heo et al., 2020)	Using Hybrid Sense Prediction	The accuracy proposed model is 64.75.	✓ The next researcher would be expanded to include multimodal data such as photos and audio.
4.	word sense disambiguation for Tigrigna language using a semi-supervised machine	semi-supervised model	The average performance of the AD Tree, AdaBoostM1, and bagging algorithms was 93.6636	✓ Other researchers have formed a team with Tigrigna experts to produce an effective stemmer for

	learning approach (Muruts, 2018)		percent, 85.35 percent, and 79.1917 percent, respectively, while the SMO and Nave algorithms performed at 91.9224 percent and 70.8968 percent, respectively.	the Tigrigna language. ✓ Only focus on modeling WSD with lexical ambiguity.
5.	Towards the Sense Disambiguation of Afan Oromo Words Using Hybrid Approach (Unsupervised Machine Learning and Rule-Based) (W. Tesema et al., 2017)	Unsupervised Machine learning and Rule-Based	The accuracy of unsupervised machine learning is 56.2% and the hybrid approach is 65.5%.	✓ The hybrid technique is suitable for Ethiopian languages with limited resources, such as Afan Oromo. Because there is no annotated corpus, a hybrid method is essential for disambiguation.
6.	Graph Convolutional Network for Word Sense Disambiguation (Zhang et al., 2021)	based on Graph Convolutional Network (**GCN**)	WSD performance is good without too many convolutional layers, with GCN (4) =0.734 accuracy.	✓ Adding more linguistic knowledge to the WSD graph in the future and using the consideration process to improve WSD speed.
7.	An unsupervised method for word sense disambiguation (Rahman and Borah, 2021)	Unsupervised ML techniques	The evaluation is clear that the proposed method works better than existing methods. The	✓ For future work, extend the proposed method to the multilingual system.

			proposed method is 76.80%.	
8.	Word Sense Disambiguation for Wolaita Language Using Machine Learning Approach (Tadesse, 2021)	Machine learning	Using 10-fold cross-validation, the Support Vector Classifier and Bagging classifiers with TF-IDF achieved an accuracy of 83.22 percent and 82.82 percent, respectively, on WS11 (5-5).	✓ Do more than five ambiguous words, Knowledge-Based WSD for the Wolaita language using deep learning methods.
9.	Sense disambiguation for Punjabi language using supervised machine learning techniques (Singh and Kumar, 2019)	Using Supervised ML and deep learning classifiers	In 300-dimensional Fast Text word embedding, the accuracy of SVM is 83.75 and LSTM is 83.45.	✓ This is a single target word WSD system that should be expanded to an all-word WSD system.

10.	Word Sense Disambiguation Using Prior Probability Estimation Based on the Korean WordNet (Kim and Kwon, 2021)	Using unsupervised disambiguation and (using a knowledge-based approach).	The average lexical disambiguation accuracy was 86.3 percent, which was about 4% lower than when SENSEVAL2 data was used.	✓ Increase the system's reliability by evaluating additional ambiguous terms using other assessment data. ✓ A study on preprocessing, such as selection limitations, would be conducted for an analysis that cannot be solved by statistical information due to a data scarcity problem.

In this study, supervised ML techniques would be used for a selected research approach; it is used as a source of information for disambiguation. The above-summarized research papers try to fill some research gaps that have been done on Cushitic languages and other low-resource languages. Review the above-published papers based on my interests and by analyzing the research gaps of each paper. The following gaps have been addressed in this study:

- ✓ In the above papers on low-resource Ethiopian languages like Afaan Oromo, the researcher also recommends for future work for other languages to use the hybrid model, and I am using this model for the Hadiyyisa Language.

CHAPTER THREE

3. MATERIALS AND METHODS

3.1. Overview of Research Methodology

This chapter discusses the principles of research materials, techniques, and methodology to gain a thorough understanding of the research methodology used in past research as well as our current research study. This chapter is mainly focused on the basic data collection mechanisms; data sources; data preparation procedures; feature extraction methods; and model evaluation were primarily addressed. This study focused on supervised machine learning methods

Supervised methods: In this type of method, annotated corpora are used to train ML, but a problem may arise that such corpora are very time-consuming to create. A training set of feature-encoded inputs and their associated sense label category are included in the supervised learning system (Pal and Saha, 2015)(Han and Shirai, 2021) (Alemu and Fante, 2021)(Kang et al., 2018). One of the more difficult aspects of WSD is the annotated corpus or context-based repository technique. For that reason, data has been collected from different data sources, including the Hadiyyisa Bible, HTV, Institution of Radio, Hadiya Zone Education Bureau, and Wachemo University teaching materials. This study used supervised ML methods because it is sufficient to handle resource-scarce language.

Optimization techniques are used to improve model prediction accuracy. These optimization techniques are:-SGDClassifier for SVM, MLPClassifier for NN, Voting Classifier for Hybrid models(NB-NN, SVM-NN, SVM-NB, and SVM-NB-NN), and Voting Classifier also for Ensemble machine learning(DT-RF, DT-GB, RF-GB, and DT-RF-GB). These optimization classifiers have been briefly discussed in the below sub-section.

SGDClassifier

Stochastic Gradient Descent (SGD) is a simple yet efficient optimization algorithm used to find the values of parameters or coefficients of functions that minimize a cost function. This classifier implements a plain SGD learning routine that supports different loss functions and penalties for classification. It is used to find the model parameter that corresponds to the best fit between predicted and actual results. It is also one of the optimization mechanisms.

From these hyperparameters in this study, the following parameters were affecting the model accuracy. These are **loss, random_state, max_iter, tol, alpha,** and **penalty.** Here are applied to optimize the model accuracy. The main advantages of the SGD classifier are that they are

very efficient and very easy to implement, as there are many opportunities for code tuning. Its drawbacks require several hyperparameters like regularization parameters.

MultinomialNB Classifier

This classifier algorithm is widely used for assigning documents to classes based on the statistical analysis of their contents. It provides an alternative based on semantic analysis and simple textual data classification. In this classifier, use the **alpha** parameter that controls the form of the model itself. This **alpha** parameter should be applied to increase the accuracy of the correct prediction.

MLPClassifier

It can also have a regularization term added to the loss function that shrinks model parameters to prevent overfitting. They are different types of solvers for weight optimization.

- ✓ '**lbfgs'** is an optimizer in the family of quasi-Newton methods.
- ✓ '**sgd'** refers to stochastic gradient descent.
- ✓ '**adam'** refers to a stochastic gradient-based optimizer.

The **lbfgs** optimizer is used in our research study. The default solver, **adam**, works pretty well on relatively large datasets (with thousands of training samples or more) in terms of both training time and validation score. For small datasets, however, **lbfgs** can converge faster and perform better. The **relu** activation function provides better results in increasing the accuracy of the model than other activation functions like softmax, logistic, and Tanh.

Voting Classifier

It is a machine learning model that trains on an ensemble of numerous models and predicts an output (class) based on the highest probability of the chosen class as the output. It simply aggregates the findings of each classifier passed into the Voting Classifier and predicts the output class based on the highest **majority of voting**. The idea is that instead of creating separate dedicated models and finding the accuracy for each of them, to create a single model that is trained by these models and predicts output based on their combined **majority of voting** for each output class.

DecisionTreeClassifier

It is a supervised machine learning model. This means that they use pre-labeled data to train an algorithm that can be used to make a prediction. To reduce memory consumption, the complexity and size of the trees should be controlled by setting those parameter values.

RandomForestClassifier

It is supervised machine learning, which you can use for classification problems. It is among the most popular machine learning algorithms due to its high flexibility and ease of implementation.

GradientBoostingClassifier

It builds an additive model in a forward-stage-wise fashion; it allows for the optimization of arbitrary differentiable loss functions.

3.2. Research Design

The overarching research policy provides a concise and reasonable plan to address the well-known research question(s) through data collection, interpretation, analysis, and discussion. The methodology and procedures used in the research design, as well as the framework of

research methods and techniques used to conduct the suggested study.

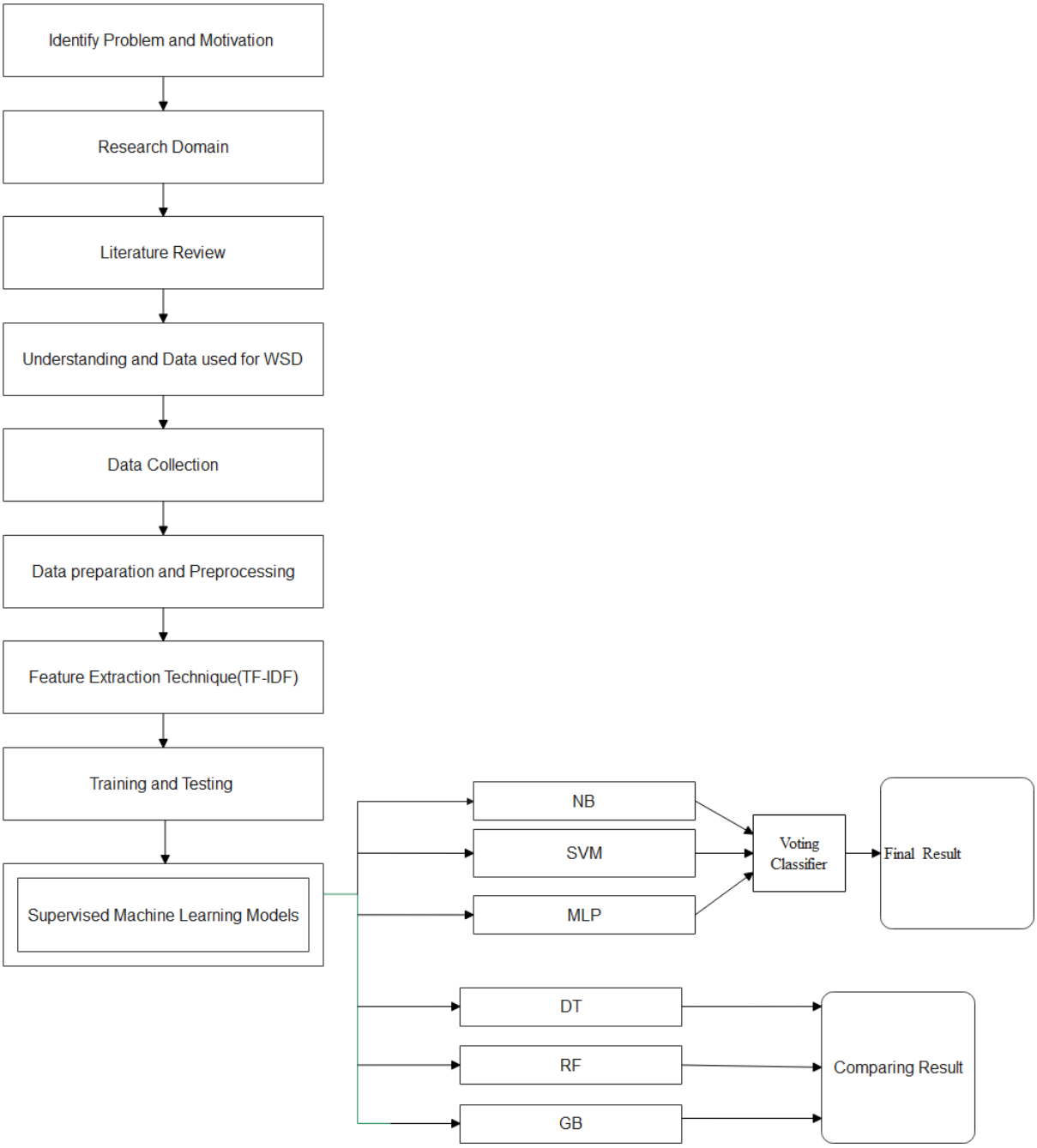


Figure 3. 1 Research Design Diagram

3.3. Data Collection and Data Preparation

3.3.1. Sources of Data

In this study, two main techniques were used to collect and survey complete information about ambiguous word selection and corpus preparation. Such data collection mechanisms are primary data collection and secondary data collection methods.

Primary Data Collection: To collect the data by interviewing the necessary people about WSD in selected domain areas such as HTV, Wachemo University, Hadiyyisa Department, Hadiya Zone Education Bureau, and Hadiya Zone Culture and Tourism Bureau. In this method, different NLP researchers on how to collect and design effective WSD corpus preparation were interviewed. With this data collection method, I interviewed two native Hadiyyisa speakers about ambiguous words, which most people probably use for different communication purposes.

Secondary Data Collection: was collected from an offline and online textbook, related literature review, appropriate research methods, academic documents, and student learning textbooks; from these data sources to collect academic data from the Hadiyyisa Language and Literature Department modules and primary textbooks. Offline data sources include academic books (grade 1–12 Hadiyyisa language textbooks and modules of higher education), spiritual offline materials, government office norms, and guiding principles. [3] Online data sources are used to acquire data that was previously stored on an internet domain and was freely available. Researchers got this information from free and open-source MT platforms. The Hadiya Zone Cultural Bureau and Tourism Office have selected ambiguous words that the majority of people are likely to use for genuine conversation. There is no dataset in my thesis work, and even if I continued to search the internet, no one published a paper or provided a resource for WSD for the Hadiyyisa language. So, as a baseline paper, I considered the various local under-resourced languages was seen.

3.3.2. Annotation of Data

Data annotation is the process of classifying text data. It is one of the challenging areas of this study.

[3] Hadiyya-Engl-Dict-2007.10-short-form.pdf (hadiyajourney.com)

Table 3. 1 Data Annotation Result for Word Sense Disambiguation for the Hadiyyisa Language

No	Ambiguous words	Sense 1	Sense 2	Total senses
1.	Anga(n)	Haraa'mmima(favour) = 138	Qaamafeeta(hand)=184	322
2.	Misha(n)	Hamaama haqqi misha(Fruit)= 116	Siixo'o tie'matooota(Product or result)=154	270
3.	Diinate(n)	Mine Hoora(domestic animals)= 144	Birra (Money)= 126	271
4.	Hurbaata(n)	Ichcha(food)=222	Sire'e(Seed)=135	357
5.	Hagara(n)	(colour) moo'akkam haga'l = 126	qocca baxo (type, kind) =132	258
6.	Seera(n)	Wi'lonne hawwone hara'manacha(Help organization) = 132	Sono'o, ogora(rule, law)=204	336
7.	Inqe(n)	iicoo qoxxaalli orachcho= 120	Wo'ma= 104	224
8.	biqe(n)	orachchi baxxancha= 139	Sagada= 128	267
9.	Ille(n)	mooimmina hasisoohane =119	foorami wosha sentence count= 135	254
10.	Bu'o(n)	shooto'o =139	itakaammi waasi fiiro ichcha hurbaata =161	300
				2,872

3.3.3. Dataset Preparation

The size of the data collection influences the quality of the disambiguated senses by considering the general rule that more data is better (Tesema, 2016)(Eshetu et al., 2020). By

following the procedures in this research, how to prepare the dataset for the new WSD for the Hadiyyisa language has been prepared.

Context

There are now only a few keep-fit options available for analyzing WSD approaches. Thes e data collections have been created to make the evaluation technique easier and faster.

2,872 sentences have been collected, each ambiguous word with two polysemy words.

Content

The dataset is prepared in an excel spreadsheet (.xlsx). It contains four columns: serial number (SN), sentence/context, polysemy word, and sense. The sentences are in the second column (context). There could be more polysemy terms in the statement. As a result, the target word (polysemy term) in the third column is required in the fourth column. Use the sentence to see if your WSD system can distinguish the target word in the fourth column from the equivalent sentence in the second column.

3.4. Data Analysis

Depending on the study scope, various kinds of ambiguity may arise in the suggested investigations into NLP: -lexical and phonological (sound) ambiguity, as well as referential and semantic ambiguity (Assemu, 2011). This research study is focusing on lexical ambiguity.

3.4.1. Data Preprocessing

Preprocessing: Tokenization, character normalization, stop word removal, and punctuation mark removal were used for the data preprocessing phases [4](Han and Shirai, 2021)(Rahman and Borah, 2021)(Eshetu et al., 2020). This data preprocessing phase has been applied to this study to improve the accuracy of model performance to predict and detect the correct sense of words.

Data Cleaning: Data cleaning removes extraneous information that could affect the model's performance. Special letters, punctuation, and symbols have been commonly seen in data acquired from many sources. This has been cleaned up by using data preprocessing techniques to remove any unnecessary features and used to improve model

[4] 11 Techniques of Text Preprocessing Using NLTK in Python - MLK - Machine Learning Knowledge

accuracy (Tadesse, 2021)(Eshetu et al., 2020). Pseudo code for Data cleaning for Hadiyyisa text seen below as follows:

Input: Uncleaned Hadiyyisa corpus

Output: Cleaned Hadiyyisa corpus

Step 1: Start or initialization

Step 2: Read the unclean corpus

Step 3: If the corpus contains punctuation marks then

A) Remove the punctuation marks.

B) Display a cleaned corpus without punctuation marks.

Step 4: Else if the corpus contains a sentence

A) Tokenize the corpus into words

B) Display the tokenized data set

Step 5: Else if the corpus contains stop words

A) Remove stop words from the tokenized corpus

B) Display without stop words

Step 6: Display cleaned corpus

Step 7: Stop

Tokenization

It is the process of segmenting a string of characters into words. It divides the raw text into tokens, which are words or sentences. It is used to deduce the meaning of a text by looking at the order of the words (Woldeyohanes, 2016). This is an example of a tokenization process using the NLTK Toolkit (python). For example, let us see Ergoge waasa hurbaata ito'o. This sentence is tokenized as follows {'Eregoge,' waasa,'hurbaata,' itoo'}.

Stop word Removal

They are any words in a stop list (or negative dictionary) that are filtered out before or after natural language data (text) processing and are often utilized in natural language writings (Eshetu et al., 2020). The most common words in any language (articles, prepositions, pronouns, conjunctions, and so on) do not add anything to the text. Because there are fewer tokens involved in the training, removing stop words minimizes the dataset size and hence reduces training time. Removing extraneous terms from the dataset increases model performance and is essential for NLP model training. This study used the sentences-based window size to load sentences from the prepared dataset (Woldeyohanes, 2016) (Raulji, 2016) (Faisal, 2018). The stop word lists are saved in a file (**hadiyyisa.txt**), which is the most

frequent word. Some of the Hadiyyisa stop words are listed in **Annex C.** The Stop word removal algorithm is implemented as follows.

Step 1: Read the file from the corpus.

Step 2: If the term in the file is in hadiyyisa.txt, remove it Go to step 1.

Step 3: Else end process.

Normalization

In NLP, normalization is the process of translating texts into a standard format. Normalization involves the process of handling different writing systems. In this study, mainly every term in the document is converted into a similar case format, which is into a lower case (Woldeyohanes, 2016). The form of indexed text and query phrases must be the same M.K. and MK, for example, should be matched. The goal of normalization in this situation is to make the words in this corpus similar in diverse cases (Tesema, 2016). The normalization algorithm is presented as follows (Woldeyohanes, 2016)

Step1. Read files from the corpus

Step2. If the word is written in upper case change it into lower case and go to step1

Step3. Else if the word is written in lower case put the word and go to step 1

Step4. Else end process

Punctuation Mark Removal

This is a crucial data preparation step because it adds no value to the data. Punctuation (or interpunction) is the use of spacing, conventional signs (called punctuation marks), and some typographical elements to promote comprehension and accurate reading of the written text, whether read silently or aloud. "It is the practice, action, or system of inserting points or other small marks into texts to aid comprehension; the use of such marks to divide the text into sentences, phrases, and other units." (Woldeyohanes, 2016)(Faisal, 2018)[5].

3.4.2. Feature Extraction Techniques

Why do we need feature extraction techniques?

Machine learning algorithms provide output for the test data using a pre-defined set of features from the training data[6]. The fundamental issue is cannot work directly on raw text (Papandrea et al., 2017). To transform the text into a matrix (or vector) of features, we

[5] Punctuation - Wikipedia

[6] Feature Extraction Techniques - NLP - GeeksforGeeks

would need some feature extraction techniques. The following are the most prevalent feature extraction methods: Bag-of-Words and TF-IDF.

Bag of Words: this is a straightforward method for converting tokens into a set of features. This model is used in document classification to train the classifier using each word as a feature.

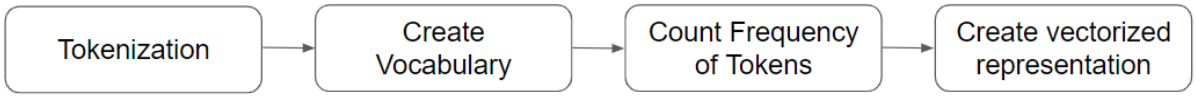


Figure 3. 2 Procedures to Create Bag of Words[7]

To create BoW, first understand the basic pseudo-code algorithms shown below as follows:-

Step 1: Format the full text.

Step 2: Perform text preparation, which involves converting the entire text to lowercase letters and deleting all punctuation and unnecessary symbols.

Step 3: From the prepared corpus, create a vocabulary of all unique terms.

Step 4: Create a feature matrix by giving each word a column and assigning each row to a review.

Step 5: Perform text vectorization. For step 4 if there is text present, otherwise if there is no text.

The biggest disadvantage of this strategy is that the sequence in which words appear is lost; instead, develop a randomized token vector. Therefore, instead of individual words, consider N-grams (mainly bigrams) to tackle this problem (i.e., unigrams). This preserves the local word order (Tadesse, 2021).

TF-IDF Vectorizer

TF-IDF stands for term frequency-inverse document frequency. The TF–IDF score increases with the number of times a word appears in the document and decreases with the number of documents in the corpus that contain the word. TF-IDF means a near algorithm that uses the occurrence of words to define how essential those words are to a given document. It has been divided into two sections:

1. Term Frequency (TF)
2. Inverse Document Frequency (IDF)

Term Frequency (TF)

[7] https://d1rwhvwstyk9gu.cloudfront.net/2020/08/Approach_bag_of_words-1.png

Term frequency refers to how many times a term appears in the document. It might be compared to the likelihood of discovering a word within a document. It is written like this:

$$TF_{(t,d)} = \frac{n_{t,d}}{n_d} \quad (3.1)$$

where $TF_{(t,d)}$ is the number of terms t in the document $n_{t,d}$ is the number of occurrences of term t in a document? whereas n_d is the sum of the terms in document d (Woldeyohanes, 2016)(Faisal, 2018)(Tadesse, 2021).

Inverse Document Frequency (IDF)

The inverse document frequency determines whether a term is rare or common over the entire corpus of texts. It emphasizes phrases that exist in a small number of texts across the corpus or words with a high IDF score in plain English. Divide the total number of documents in the corpus by the number of documents containing the phrase to get the logarithm of the overall word (Woldeyohanes, 2016)(Faisal, 2018).

$$idf(t) = \log(N / (df + 1)) \quad (3.2)$$

The value of IDF (and consequently TF-IDF) is greater than or equal to zero since the ratio inside the IDF's log function must always be greater than or equal to one. The ratio inside the logarithm approaches one when a phrase appears in a high number of documents, and the IDF approaches 0. The product of Term Frequency and Inverse Document Frequency (TF-IDF) is

$$TF - IDF_{(t,d)} = TF_{(t,d)} * IDF(t,d) \quad (3.3)$$

The TF-IDF score of a phrase with a high frequency in a document but a low document frequency in the dataset is high. For a word that appears in almost all publications, the IDF value approaches 0, pushing the TF-IDF closer to zero. The TF-IDF value is high when both the IDF and TF values are high, indicating that the phrase is uncommon within the document yet prevalent within it (Faisal, 2018). For example, let us consider three of these reviews.

Sentence 1:- ise hurbaata ito’o

Sentence 2:- Hurbaata ito’o

Sentence 3:- ise hurbaata saato’o

Consider all the unique words from the above set of reviews to create a vocabulary, which is going to be as follows:-{Hurbaata, ise, itoo, saatoo}.

Table 3. 2 Examples for Term Frequency-Inverse Document Frequency

	Hurbaata	Ise	Ito'o	Saatoo
Sent-1	1*log(3/3)+1=1	1*log(3/2)=0.17	1*log(3/2)=0.17	0
Sent-2	1*log(3/3)+1=1	0	1*log(3/2)=0.17	0
Sent-3	1*log(3/3)+1=1	1*log(3/2)=0.17	0	1*log(3/1)=0.47

As a result, my contributions in this part are comparative literature for feature extraction in foreign and local languages that is currently more widely used and the construction of an acceptable method for Hadiyyisa text.

Now choosing TF-IDF is an appropriate feature extraction technique for these Hadiyyisa texts, because it fills the shortcomings of the Bag of Words and it is faster than BoW. However, a dataset was prepared at the sentence level to calculate each term's highest frequency in the sentence. Therefore, TF-IDF is an appropriate feature extraction technique to convert the text file into a machine-readable format.

3.4.3. Model Selection

The main purpose of this study is to build a WSD model for the Hadiyyisa language using supervised machine learning techniques. Due to the lack of Hadiyyisa WordNet and annotated datasets, this study has focused on ten ambiguous words that mean lexical sample training of WSD. Supervised machine learning methodologies have been used for the suggested research to customize the following models: Neural Network, Support Vector Machine, Nave Bayes, Decision Tree, Random Forest, Gradient Boosting, and Hybrid models. In this study, after concluding the result of the experiment, selected only two models such as **SVM** and **NB-SVM-NN**, which are appropriate to handle and achieve the best results of WSD for the Hadiyyisa language.

Support Vector Machine

There are several classification algorithms used in machine learning. However, SVM is better than most of them since it produces more accurate results. For that case, here selected for this study to perform the SVM Classifier it has lower computational complexity than other classifiers, and it should be used even if the number of both senses of instances is not equal because it can normalize the data into the space of the decision border separating the two classes. In addition, SVM performs admirably when there is a clear separation between

classes. The main advantage of the SVM classifier is the high quality of the classification results that have been achieved (Demidova et al., 2019).[8]

Hybrid Models

This hybrid model includes all deterministic rules, allowing us to train it in the same way as any other machine-learning model. Fortunately, Scikit-learn provides a Base Estimator class that can inherit to quickly create our Scikit-learn models. Only slight gains in prediction accuracy are seen by hybrid approaches (SVM, NN, NB) when compared to standard understandable methods and integrated by using a Voting Classifier method (Miškovic, 2014)(Salt et al., 2016). The accuracy and efficiency of a hybrid technique are to modify the SVM classifier (Demidova et al., 2019). Figure 3.3 Shows how the hybrid learning model works in full, merging all of the concepts discussed in this study into one model(Salt et al., 2016).

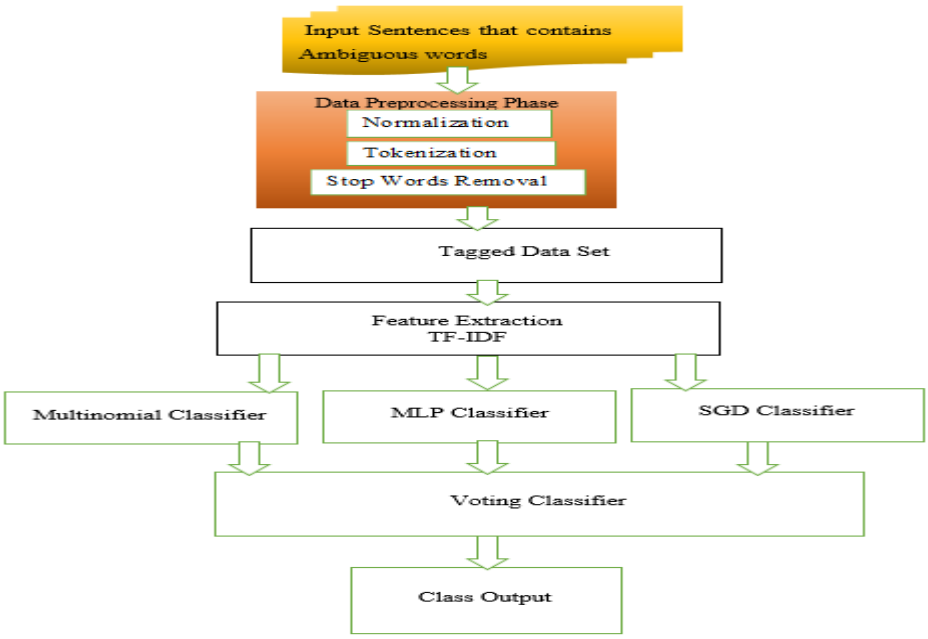


Figure 3. 3 A Hybrid Model Working Process

3.4.4. Evaluation Metrics

Accuracy, precision, Recall and F-measure have been commonly used to assess the performance of classification algorithms.

The **recall** is a percentage number that indicates how many right results have been found out of a set of results from a processed document detected (based on the expectations of a certain application).

[8] SVM Algorithm - Bing images

Recall = the number of CorrectlyDisambiguated Words / the number of Total number of Ambiguous Words. (3.4)

The **precision** is a percentage metric that indicates how many of a set of outcomes from a processed document is right (based on the expectations of a certain application).

Precision = the number of CorrectlyDisambiguated Words / the number of Disambiguated Words (3.5)

The **F-Score** is the harmonic mean of a system's precision and recall scores. Some high F-scores can be the result of an imbalance between precision and recall, according to critics of using F-score values to judge the quality of a prediction system.

$$F1-\text{Score} = 2 \text{ x } \left[\left(\text{Precision x Recall}\right) / \left(\text{Precision } + \text{ Recall}\right)\right]. \quad (3.6)$$

Accuracy means the performance factors of the number of correctly classified and incorrectly categorized cases retrieved from the output confusion matrix (Thankappan et al., 2016)(Alemu and Fante, 2021)(Faisal, 2018).

3.5. Research Materials

3.5.1. Implementation Tools and Packages used for Word Sense Disambiguation Model of Hadiyyisa Language

This study used the Python programming language for implementing the proposed research.

Table3.3 Lists of Tools with their Version and Description

Tools	Description
Anaconda 2020.11	It is a graphical user interface for Conda packages, environments, and channels written in Python.
Jupiter Notebook 6.1.4	It is a web application written in Python that generates live code, visualizations, and narrative prose.
Microsoft Excel 2016	Used for annotation and data training tasks such as cleaning, filtering, and sorting data.
MathType 6.0	It is used for writing a mathematical formula.
Microsoft Word 2016/2010	Documentation has been written with this tool.
Wonder share EdrawMax 10.5.0	Used to sketch or design, WSD suggested architecture and research design.
Mendeley-Desktop-1.19.4-win32	It is used to insert citations for references

Table 3.4 Shows that Python packages are utilized to implement the following packages for the suggested WSD model for the Hadiyyisa language. Python imports these packages, which each perform a specific function. The Scikit-learn package, for example, is used to import supervised machine learning algorithms.

Table3. 4 Lists of Packages used for Word Sense Disambiguation

Python libraries	Description
NLTK	The platform is used to work with human language data.
Pandas	Data frames have been used for reading, writing, and manipulating this program.
Numpy	is used to process multi-dimensional arrays.
Sea born	Used to visualizing data in an appealing interface.
Matplotib	It has been used to visualize data.
Gradio	Is used to build a GUI library that allows you to create customizable GUI components for Machine Learning models.
Pickle	A pickle is a database or file that stores and converts Python objects.

CHAPTER FOUR

4. PROPOSED WORD SENSE DISAMBIGUATION MODEL FOR HADIYYISA LANGUAGE

This chapter describes the proposed WSD model for Hadiyyisa language. It mainly focuses on how the corpus is prepared, design requirements, the proposed architecture for Hadiyyisa WSD model, document preprocessing techniques, preparing machine-readable datasets, and evaluation techniques of the model.

4.1. Proposed Architecture of Word Sense Disambiguation Model for Hadiyyisa Language

The proposed WSD model is depicted in Figure 4.1. It has to be noted that the model has several components for data preprocessing, training a classifier, and classifying an input sentence. The model takes sentences that contain ambiguous words as input. The sentences would be preprocessed to make them suitable for further processing. Then the classification algorithms (NB, SVM, and NN) build a classifier from the training set, apply the built classifier to test new instances, and due to that, display the performance evaluation of the classifier. A detailed explanation of the model for WSD has been briefly discussed in the below diagram. Figure 4.1 Shows the discussion about the proposed WSD model for the Hadiyyisa language. The proposed WSD model for Hadiyyisa language contains the following components which have been presented briefly as follows

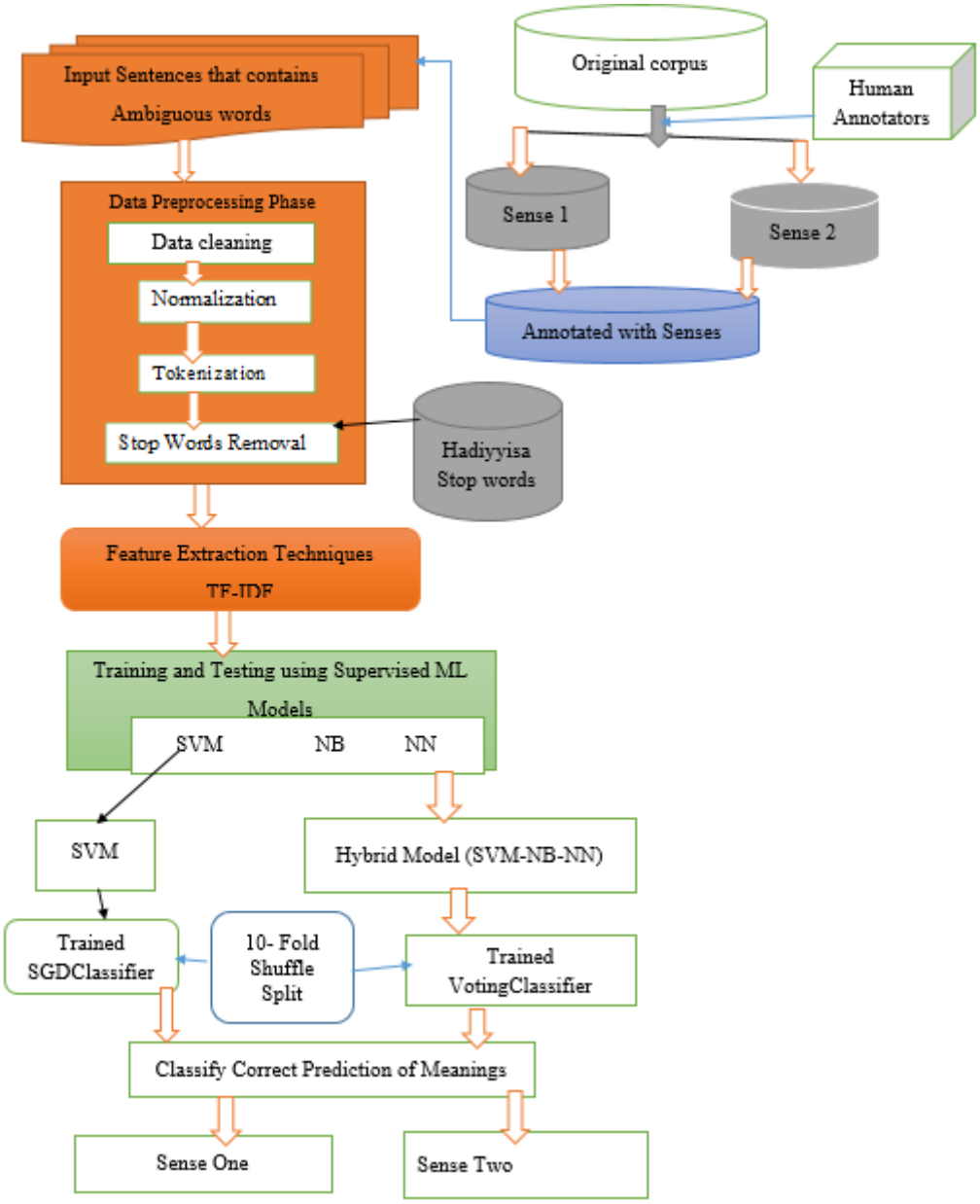


Figure 4. 1 Word Sense Disambiguation Model for Hadiyyisa Language

4.1.1. Preparation of Corpus

First, searching for ambiguous words in documents collected raw text from different data source sites. The total data of 2872 sentences that contain ambiguous words are collected from HTV, student learning textbooks, online bible Hadiyyisa books, and offline textbooks. Although there is no Hadiyyisa stop word list available for use, so I manually prepared the list using the trained data and other Hadiyyisa student textbooks. The prepared stop word list is normalized for data consistency. The following figure 4.2 shows the sample search results of actual ambiguous words in Hadiyyisa language documents.

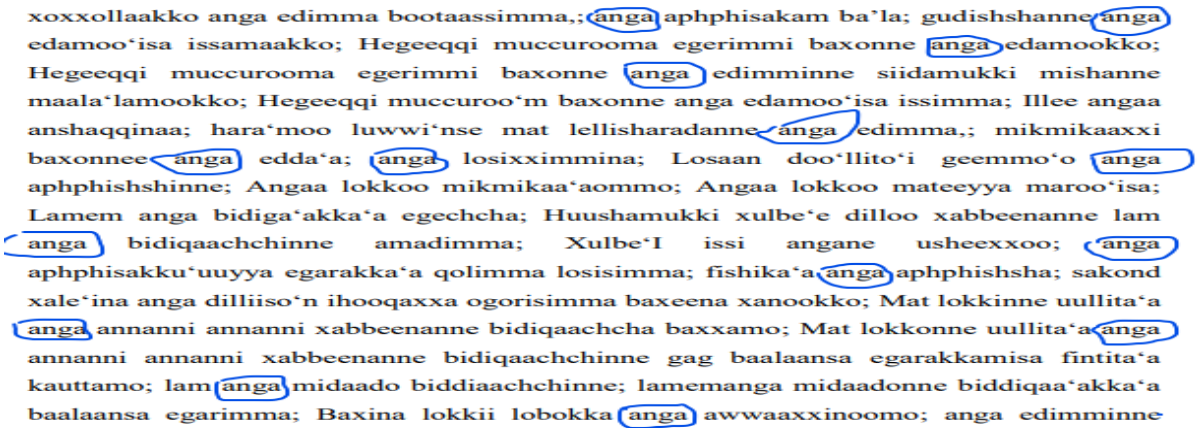

xoxxollaakko anga edimma bootaassimma,; anga aphphisakam ba'la; gudishshanne anga edamoo'isa issamaakko; Hegeeqqi muccurooma egerimmi baxonne anga edamookko; Hegeeqqi muccurooma egerimmi baxonne anga edimminne siidamukki mishanne maala'lamookko; Hegeeqqi muccuroo'm baxonne anga edamoo'isa issimma; Illee angaa anshaqqinaa; hara'moo luwwi'nse mat lellisharadanne anga edimma,; mikmikaaxxi baxonnee anga edda'a; anga losixximmina; Losaan doo'llito'i geemmo'o anga aphphishshinne; Angaa lokkoo mikmikaa'aommo; Angaa lokkoo mateeyya maroo'isa; Lamem anga bidiga'akka'a egechcha; Huushamukki xulbe'e dilloo xabbeenanne lam anga bidiqaachchinne amadimma; Xulbe'I issi angane usheexxoo; anga aphphisakku'uuyya egarakka'a qolimma losisimma; fishika'a anga aphphishsha; sakond xale'ina anga dilliiso'n ihooqaxxa ogorisimma baxeena xanookko; Mat lokkinne uullita'a anga annanni annanni xabbeenanne bidiqaachcha baxxamo; Mat lokkonne uullita'a anga annanni annanni xabbeenanne bidiqaachchinne gag baalaansa egarakkamisa fintita'a kauttamo; lam anga midaado biddiaachchinne; lamemanga midaadonne biddiqaa'akka'a baalaansa egarimma; Baxina lokkii lobokka anga awwaaxxinoomo; anga edimminne

Figure 4. 2 Searching Result

4.1.2. Annotation of Corpus

In supervised ML, the annotation of the training dataset with target word output is important. Each of the sentences in the collection is annotated according to its interpretation of the senses as well as context meanings. Then the annotation was done independently by two native speakers of Hadiyyisa, as suggested by previous researchers was verified.

4.1.3. Preprocessing

After the dataset is annotated, apply the data preprocessing phase for this tagged dataset. In this phase, different types of preprocessing like tokenization, stop word removal, normalization, and punctuation mark removal were applied (Dereje, 2017).

4.1.4. Feature Extraction

In this phase, here applied a feature extraction mechanism to this cleaned dataset to convert text into the machine-readable format by using the TF-IDF mechanism (Tadesse, 2021).

4.1.5. Model Learning

For the model here presented in Figure 4.1, the features used a sentence-based window size, which means n-gram. In this study, there has a fixed-sized dataset. We train the supervised machine learning classifier on the entire dataset, and the classifier accepts given sentences to classify them according to the classification knowledge acquired on the training set. During the experiment, the dataset is divided into a training set (80%) and a testing set (20%). To build the model for the WSD and finally, apply the supervised machine learning model and integrate different supervised machine learning models by using voting classifier techniques. Different types of supervised machine learning algorithms, such as regression and

classification algorithms. Therefore, this study is focused on classification algorithms such as SVM, NB, and NN. In addition to that, other classification algorithms such as Decision Tree, Random Forest, and Gradient Boosting are used to compare the results to the above classification algorithms.

CHAPTER FIVE

5. IMPLEMENTATION OF WORD SENSE DISAMBIGUATION OF HADIYYISA LANGUAGE

5.1. Introduction

This section presented the WSD implementation and the basic necessary steps to develop a model, from starting data loading up to the analyzing of results using evaluation measurement or metrics. The following steps are applied to perform the implementation of the proposed model. These are:

1. Loading the required libraries and modules
2. Loading of dataset
3. Data preprocessing
4. Training and Testing Dataset
5. Feature Extraction
6. Model evaluation

These six steps are briefly presented in the below sub-section.

5.2. Loading Libraries and Modules for Word Sense Disambiguation

A system for storing valuable routines used by different programs is known as loading libraries. Install the essential libraries for WSD tasks in this section and use the sample code to load libraries for the Hadiyyisa language.

```
In [1]: #Loading the required libraries and modules for word sense Disambiguation
        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.pipeline import Pipeline
        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

        get_ipython().run_line_magic('matplotlib', 'inline')
```

Figure 5. 1 Loading the Required Libraries and Modules for Word Sense Disambiguation

5.3. Loading of Dataset

The loading of datasets is critical for NLP and machine learning applications. Well-documented data collection is the most important aspect of the ML technique. It is also the data that must be loaded before any machine learning approach can begin. There was no previous data set for the Hadiyyisa language for the WSD task. As a result, attempted to prepare the ten ambiguous words dataset in Python using the pandas: pd.read_excel () function commands and store the data in a machine-readable format using Pickle libraries in this proposed study. This function is particularly adaptable for machine learning methods, and sample code implementation for dataset loading as provided below.

```
In [2]: import string
        df = pd.read_excel("wsd_1.xlsx", header = None)
        df.rename(columns={0:'sentence', 1:'target word',2:'label'}, inplace=True)
        print(df.head(8))
        my_labels = ['haraa'mmima', 'moo'akkam haga'l annannooma','ichcha','qaamafeeta','sono'o','siixo'o','sire'e','qocca baxo'
```

	sentence	target word	label
0	anga edukko	anga	haraa'mmima
1	anga edda?	anga	haraa'mmima
2	anga gaassukko	anga	haraa'mmima
3	anga haraara	anga	haraa'mmima
4	lonsoni anga hoffiixxa manna hara'mmittamo	anga	haraa'mmima
5	anga ejja	anga	haraa'mmima
6	anga baxo edukko	anga	haraa'mmima
7	anga issukko	anga	haraa'mmima

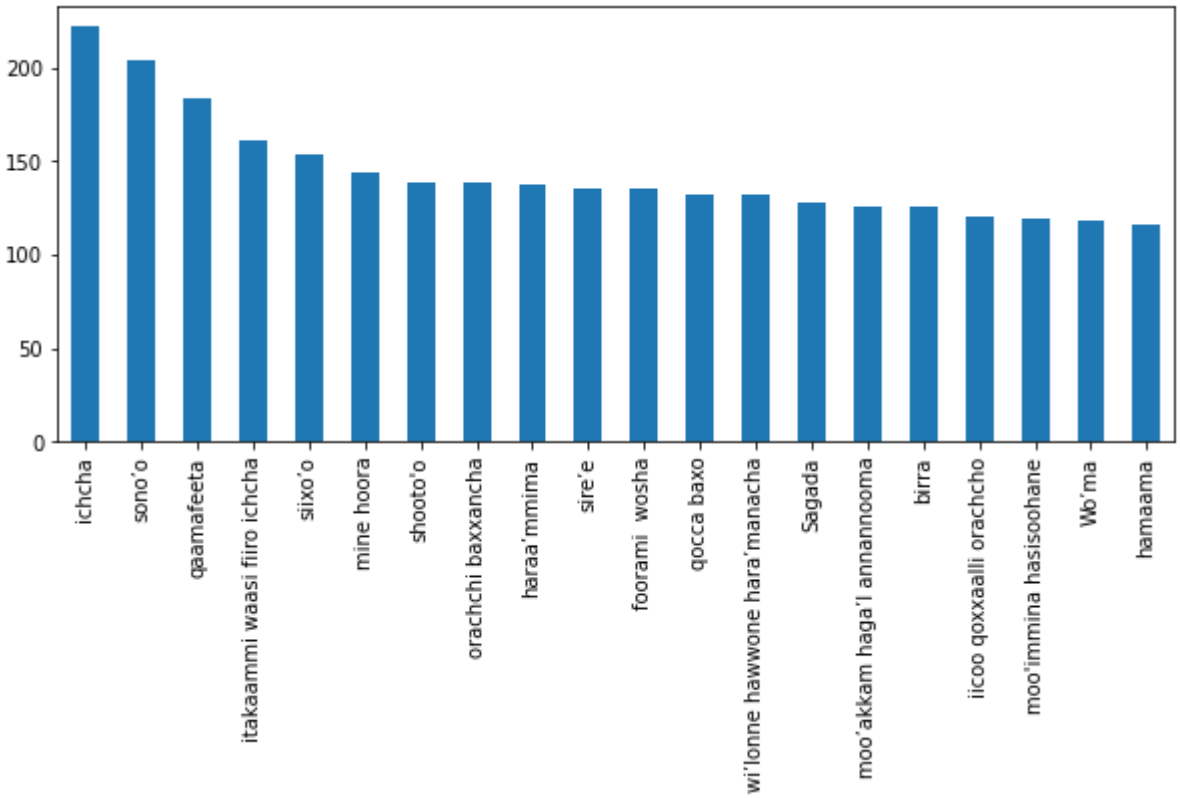


Figure 5. 2 Loading Data set for Ten Ambiguous Words with their Senses Represented by Bar Graph

5.4. Data Preprocessing for Hadiyyisa Language

This research uses the basic necessary common preprocessing phases such as removing stop words, removing punctuation, tokenization, removing numbers, and lower case letters applied to the Hadiyyisa language. Therefore, data preprocessing is very important to increase the accuracy of the model. The sample code for the data preprocessing phase is shown below.

```
In [16]: import nltk
         from nltk.corpus import stopwords
         from nltk.tokenize import word_tokenize
         import time
         from nltk import word_tokenize
         STOPWORDS =stopwords.words('hadiyyisa')
         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 hadiyyisa: hadiyyisa.translate(str.maketrans('', '', string.punctuation)))
            # hadiyyisa special unque puctuation marks different from other language
             df["sentence"] = df["sentence"].apply(lambda hadiyyisa: hadiyyisa.translate(str.maketrans('', '', "’!’‘")))
             #remove number number
             df["sentence"] = df["sentence"].apply(lambda hadiyyisa: hadiyyisa.translate(str.maketrans('', '', "0123456789")) )
             #remove stop words
             df["sentence"] = df["sentence"].str.split(' ').apply(lambda hadiyyisa: ' '.join(k for k in hadiyyisa if k not in STOPWOF
            #tokenize sentence into words
         sentence = df['sentence'].tolist()
         map(word_tokenize, sentence)
         df['sentence'].apply(word_tokenize)
         df['sentence'].apply(word_tokenize).tolist()

         df['sentence'] = df.apply(lambda column: nltk.word_tokenize(column['sentence']), axis=1)
         df['sents_length'] = df.apply(lambda column: len(column['sentence']), axis=1)

         print(df['sentence'])
         clean_text()
         df.head(5)
```

```
0                                      [anga, edukko]
1                                        [anga, edda]
2                                    [anga, gaassukko]
3                                      [anga, haraara]
4      [lonsoni, anga, hoffiixxa, manna, hara, mmittamo]
```

Figure 5. 3 Implementation Code for Data Cleaned

5.5. Training and Testing Datasets

Datasets are utilized to develop the model in the training phase, whereas data sets are applied to improve the research's overall performance in the test phase or validation phase. This traini ng and testing phase is critical in determining whether the data is either over-fitting or under-f itting. Using Scikit-learn methods like train_test_split () to train and test data points. We have trained and tested our model to assess its correctness and detection of the actual values or pre dicted values, and have divided the prepared data set into two phases: the train set and the test set phase. Therefore, in this study, to train and test the actual dataset, here used two mechanis ms to train and test the dataset. These are 80% for training and 20% for the test, 75% for train ing, and 25% for the test. In this study, to train and test the data set, which is the most appropr

iate one depending on the experimental result, was selected at **80% for the train set** and **20% for the test set.** The total dimensions of data size for our model are 2872 dimensions.

From these data points, 2297 data points are assigned to the training data set, and 575 data points are assigned to the testing dataset out of 2872 data points. The sample output for trained, tested and predicted sentences for the Hadiyyisa text is shown in **Annex D.**

5.6. Converting Text to Word Frequency Vectors with TF-IDF

In this section, to transform the raw Hadiyyisa text into a machine-readable format by using the TF-IDF feature extraction technique. Since this technique is utilized to improve the accuracy of our model, it is acceptable to handle and build WSD for the Hadiyyisa language. In this study, using TfidfTransformer in the pipeline to vectorize the text data to be trained by the model. To implement TF-IDF, using CountVectorizer as TF and IDF ignore the irrelevant features of TF. To transform data, using the **fit ()** method in Python. The implementation of TF-IDF is depicted.

```
wsd_hybrid_NB_SVM_NN = VotingClassifier(estimators)
wsd_hybrid_NB_SVM_NN = Pipeline([('vect', CountVectorizer()),
              ('tfidf', TfidfTransformer()),
              ('clf', VotingClassifier(estimators=[ ('nbs1', NB),('svm1', SVM),('nn1', NN)],weights=[1,1,1])),
             ])
wsd_hybrid_NB_SVM_NN.fit(X_train, y_train)
filename = 'wsd_hybrid_NB_SVM_NN.sav'

pickle.dump(wsd_hybrid_NB_SVM_NN, open(filename, 'wb'))
y_pred = wsd_hybrid_NB_SVM_NN.predict(X_test)
```

Figure 5. 4 Feature Extraction Using TF-IDF

5.7. Discussion of Selected Models for Word Sense Disambiguation

This section discusses the model selection mechanisms. Model selection is important to decide which model provides better performance and accuracy for WSD. Then, there are different mechanisms to select the best model for our study depending on the evaluation mechanisms such as cross-validation and average accuracy of models. This study by using supervised machine learning models and hybrid models attempted the solution for WSD in the Hadiyyisa language. Besides, from the supervised machine learning models, this study chose the following classification Algorithms:-Naive Bayes, Support Vector Machine, Neural Network, Decision Tree, Random Forest, Gradient Boosting, and Hybrid Ensemble classifier.

However, from the Classification algorithms after experimental analysis result have seen then four better result-performed models are selected for this study. Therefore, the four selected classification algorithms are discussed below briefly.

Support Vector Machine

There are several classification algorithms used in machine learning. However, SVM is better than most of them since it produces more accurate results. In that case, this study to perform the SVM classifier has lower computational complexity than other classifiers, and it could be used even if the numbers of both senses of instances are not equal because it can normalize the data or project into the space of the decision border separating the two classes. In addition, SVM performs admirably when there is a clear separation between classes. Figure 5.5 shows the sample implementation code for the SVM model.

```
In [10]: # Linear Support Vector Machine for Hadiyyisa
from sklearn.linear_model import SGDClassifier
wsd_svm = Pipeline([('vect', CountVectorizer()),
                    ('tfidf', TfidfTransformer()),
                    ('clf', SGDClassifier(loss='perceptron',l1_ratio=0.1, penalty='elasticnet',
                                          verbose=0,epsilon=1,fit_intercept=True,alpha=0.0001,shuffle=True,
                                         random_state=42, max_iter=100,tol=1.e-3)),  ])
wsd_svm.fit(X_train, y_train)
filename = 'wsd_svm.sav'
pickle.dump(wsd_svm, open(filename, 'wb'))
from sklearn.metrics import classification_report
y_pred = wsd_svm.predict(X_test)
svmaccuracy=accuracy_score(y_pred, y_test)
print("Accuracy score on training set is {}".format(accuracy_score(wsd_svm.predict(X_train),y_train)))
print('accuracy %s' % accuracy_score(y_pred, y_test))
print(classification_report(y_test, y_pred))
#Monte Carlo Cross-Validation(Shuffle Split) is a very flexible strategy of cross-validation.
#In this technique, the datasets get randomly partitioned into training and validation sets.
from sklearn.model_selection import ShuffleSplit,cross_val_score
shuffle_split=ShuffleSplit(test_size=0.1,train_size=0.9,n_splits=10,random_state=42)
scores=cross_val_score(wsd_svm,X,y,cv=shuffle_split)
print("cross Validation scores:ShuffleSplit{}".format(scores))
print("Average Cross Validation score of 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)
```

Figure 5. 5 Implementation Code for Support Vector Machine

Hybrid Models

This hybrid model includes all deterministic rules, allowing us to train it in the same way as any other machine learning model. Fortunately, Scikit-learn provides a **Base Estimator** class that can inherit to quickly create our Scikit-learn models. The hybrid model (SVM, NN, NB) means integrating different supervised machine learning models by using the Voting Classifier for each sub-model to create base estimators. The main aim of the hybrid model is to increase prediction results and to overcome one model's disadvantages. It is used to balance any bias and variance, which means balancing overfitting and underfitting models. Figure 5.6 shows the sample implementation code for the Hybrid model.

```
In [17]: #Defining Hybrid Model
estimators = []
#Defining  Naive Bayes classifiers
NB = MultinomialNB(alpha=0.0001)
estimators.append(('nbs1', NB))
#Defining Support Vector Classifiers
SVM = SGDClassifier(loss='hinge',l1_ratio=0.5, penalty='l2',alpha=0.0001, random_state=42, max_iter=1000,tol=0.001)
estimators.append(('svm1', SVM))
# Neural Network  Classifier
NN=MLPClassifier(alpha=0.0001, max_iter=1000,hidden_layer_sizes=(6,), random_state=42,
                 solver='lbfgs')
estimators.append(('nn1', NN))
# Defining the hybrid model
wsd_hybrid_NB_SVM_NN = VotingClassifier(estimators)
wsd_hybrid_NB_SVM_NN = Pipeline([('vect', CountVectorizer()),
                 ('tfidf', TfidfTransformer()),
                 ('clf', VotingClassifier(estimators=[ ('nbs1', NB),('svm1', SVM),('nn1', NN)],weights=[1,1,1])),
                ])
wsd_hybrid_NB_SVM_NN.fit(X_train, y_train)
filename = 'wsd_hybrid_NB_SVM_NN.sav'

pickle.dump(wsd_hybrid_NB_SVM_NN, open(filename, 'wb'))
y_pred = wsd_hybrid_NB_SVM_NN.predict(X_test)
print("Accuracy score on training set is {}".format(accuracy_score(wsd_hybrid_NB_SVM_NN.predict(X_train),y_train)))
```

```
#Confisuin matrix
hybrid_NB_SVM_NN=accuracy_score(y_pred, y_test)
print('accuracy %s' % accuracy_score(y_pred, y_test))
print(classification_report(y_test, y_pred))
#Monte Carlo Cross-Validation(Shuffle Split) is a very flexible strategy of cross-validation.
#In this technique, the datasets get randomly partitioned into training and validation sets.
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_hybrid_NB_SVM_NN,X,y,cv=shuffle_split)
print("cross Validation scores:ShuffleSplit{}".format(scores))
print("Average Cross Validation score of 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)
```

Figure 5. 6 Implementation Code for Hybrid Model

Random Forest

It is an ensemble learning method for the classification of tasks at training time. It builds a decision tree on different samples and takes their majority vote for classification.

Table 5. 1 Random Forest Hyperparameters Analysis Results

Hyperparamenter	Default value	Validation curve	Grid search	Iteration value	**Optimum iteration value**
Max_depth	None	15	15	20	20
Min_samples_leaf	1	1	1	1	1
Min_samples_split	2	5	2	2	3
n_estimation	10	750	500	500	500
Accuracy	0.83	0.846	0.848	0.8329	**0.8539**

```
In [9]: # Random Forest  Classifier
from sklearn.ensemble import RandomForestClassifier

rf = Pipeline([('vect', CountVectorizer()),
               ('tfidf', TfidfTransformer()),
               ('clf', RandomForestClassifier(random_state=42,n_estimators=500,min_samples_leaf=1,
                                              min_samples_split=3,max_depth=20,class_weight='balanced')),
              ])
rf.fit(X_train, y_train)
filename = 'rf.sav'
pickle.dump(rf, open(filename, 'wb'))
from sklearn.metrics import classification_report
y_pred = rf.predict(X_test)
rfaccuracy=accuracy_score(y_pred, y_test)
print('accuracy %s' % accuracy_score(y_pred, y_test))
print(classification_report(y_test, y_pred,target_names=my_labels))
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,vmax=10)
```

Figure 5. 7 Implementation Code for Random Forest Classifier

Hybrid Ensemble machine learning

The Ensemble method combines several decision trees to produce better predictive performance than utilizing a single decision tree. The main principle behind the ensemble model is that groups of weak learners come together to form strong learners. Single

classifiers can be combined with different strategies: majority voting, probability mixture, rank-based combination, and AdaBoost. Other ensemble techniques, such as weighted voting and maximum entropy combination. This study uses **weighted voting** and **majority voting.** These simple statistics combine the predictions from multiple models. This involves fitting multiple different model types on the same training dataset, then calculating the average prediction in the case of the class label or sense with the most votes for classification. This is known as **hard voting**. Besides, voting can also be used when predicting the probability of class labels on classification problems by adding predicted probabilities and selecting the label with the largest added probability, known as **soft voting**. It is preferred when the base models used in the ensemble natively support predicting sense probabilities as it can result in better performance. Because the main reason for using ensemble models over a single model is that they can reduce the variance of the predictions and achieve better results than single models. Therefore, the voting ensemble is found in Scikit-learn via the Voting Classifier. Figure 5.8 Shows the sample implementation code for the Hybrid Ensemble machine learning model.

```
In [21]: # create the sub-models
estimators = []
#Defining decision Tree classifier
D=DecisionTreeClassifier(random_state=42,min_samples_split=3,max_depth=100)
estimators.append(('d1',D))
#Defining  Naive Bayes classifier
rf = RandomForestClassifier(random_state=42,n_estimators=500,min_samples_leaf=1,min_samples_split=3,max_depth=20,
                            class_weight='balanced')
estimators.append(('rf1', rf))
#Defining Support Vector Classifier
GB = GradientBoostingClassifier(random_state=42,n_estimators=500,min_samples_split=3)
estimators.append(('gb1', GB))
# Defining the hybrid model
DT_RF_GB = VotingClassifier(estimators,voting='hard')
DT_RF_GB = Pipeline([('vect', CountVectorizer()),
                     ('tfidf', TfidfTransformer()),
                     ('clf', VotingClassifier(estimators=[ ('rf1', rf),('gb1', GB),('d1',D)],weights=[1,1,1],voting='hard')),
                    ])
DT_RF_GB.fit(X_train, y_train)
filename = 'DT_RF_GB.sav'
pickle.dump(DT_RF_GB, open(filename, 'wb'))
y_pred = DT_RF_GB.predict(X_test)
#Confisuin matrix
hybrid_DT_RF_GB=accuracy_score(y_pred, y_test)
print('accuracy %s' % accuracy_score(y_pred, y_test))
print(classification_report(y_test, y_pred))
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(DT_RF_GB,X,y,cv=shuffle_split)
print("cross Validation scores:ShuffleSplit{}".format(scores))
print("Average Cross Validation score of 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,vmax=18)
```

Figure 5. 8 Implementation Code for Hybrid Ensemble Machine Learning Model

5.8. Model Evaluation

To evaluate the models, this study uses a Monte Carlo cross-validation technique, which is a technique for evaluating supervised and Hybrid machine learning models on the subsets of available input data and evaluating them on the complementary subset of the data. Because the main reason for using Monte Carlo cross-validation is that it is a very flexible strategy of cross-validation techniques to validate data sets randomly partitioned into training and testing sets. Cross-validation was used to detect over-fitting, which is when a pattern fails to generalize. However, we compared the Monte Carlo cross-validation 5-fold and 10-fold cross-validation techniques. Monte Carlo 5-fold cross-validation has been found to have better-predicted results than Monte Carlo 10-fold cross-validation techniques to validate the models. Therefore, overall performance can be verified using the following evaluation metrics: precision, recall, f1-score, and accuracy. Figure 5.9 shows the sample implementation code for both cross-validation techniques.

```
In [23]: from sklearn.model_selection import cross_val_score
         from sklearn.model_selection import KFold
         cv=KFold(n_splits=10,random_state=42,shuffle=True)
         score=cross_val_score(wsd_svm,X,y,cv=10,n_jobs=-1)
         print("cross validation scores:",score)
         print("Average Cross Validation score of k_fold:{}".format(score.mean()))
         #Monte Carlo Cross-Validation(Shuffle Split) is a very flexible strategy of cross-validation.
         #In this technique, the datasets get randomly partitioned into training and validation sets.
         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_svm,X,y,cv=shuffle_split)
         print("cross Validation scores:ShuffleSplit{}".format(scores))
         print("Average Cross Validation score of ShuffleSplit:{}".format(scores.mean()))
```

Figure 5. 9 Sample Code for Cross Validation

5.9. Prototype Design of Word Sense Disambiguation Model

In this study, developed prototype of the word sense disambiguation model using a Python web framework known as Gradio. However, Figure 5.11 Shows the implementation code of the graphical user interface for the Hybrid (NB-SVM-NN) model. The main goal of designing a user interface is to test our model using trained and tested data. The system prototype design of the WSD model is depicted in Figure 5.10. To design a system prototype for the Hadiyyisa language using the gradio web application in Anaconda navigator. After the data preprocessing, training and testing datasets, feature extraction, and model evaluation

have been completed. Then, the model takes sentences that contain ambiguous words as input. Then the model displays the predicted sense of ambiguous words. Figure 5.10 shows the user interfaces of the WSD for the Hadiyyisa language.

```
import gradio as gr
import pickle
def VotingClassifier(X_test):
    wsd_hybrid_NB_SVM_NN = pickle.load(open('wsd_hybrid_NB_SVM_NN.sav', 'rb'))
        X_test.lower()
        X_test=[X_test]
        X_test==y_pred[:1]
        X_test=='wsd_hybrid_NB_SVM_NN.sav'
        y_pred=wsd_hybrid_NB_SVM_NN.predict(X_test)
    return y_pred
app = gr.Interface(
    VotingClassifier,
    inputs=[
        gr.Text(value= df['sentence'].all(), label=" Lami Tirato'o Uwwo Sagara Amadaako xuunisaami Woca Agise(USer Inputs Sente
    ),
    ],
    outputs=[gr.Text(value=df['label'].all(),label="Tirato'o Uwwe(Predicting Correct Sense)")],

)

app.launch()
```

Figure 5. 10 Implementation Code for User Interface

```
Running on local URL:  http://127.0.0.1:7883/

To create a public link, set `share=True` in `launch()`.
```

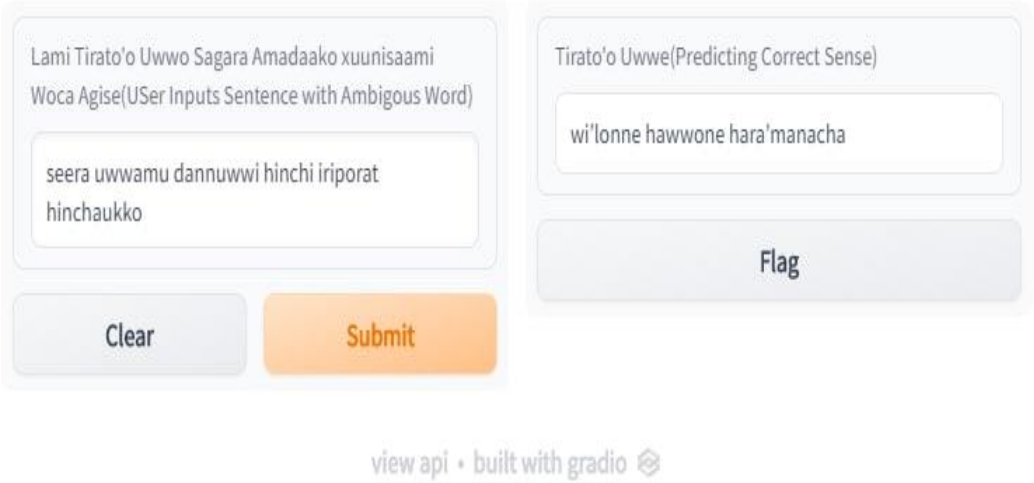


Figure 5. 11 User Interface Design for WSD

CHAPTER SIX

6. RESULT AND DISCUSSIONS

6.1. Introduction

As discussed in the above chapters, supervised word sense disambiguation has been selected for this study. Then the experimental findings are mainly examined in this chapter that focuses on the following subsections: train-test split for evaluating supervised machine learning models; parameter evaluations; The performance is evaluated with their accuracy, precision, recall, and f1-score including the confusion matrix considered for accuracy comparison; and comparing this study outcome with the existing researchers' results. However, different types of parameters are mainly important to improve the accuracy and efficiency of the model. Besides, the best-performed models have been examined with their necessary evaluation mechanisms. Generally, the research questions, their model accuracy results, evaluation metrics, and model parameter optimizer have been verified.

6.2. Discussions

6.2.1. Evaluation Procedures

The experiment was done to measure the overall performance of the developed supervised machine learning WSD model. Out of the available corpus, the training set consists of **2,297** sentences, and the remaining **575** sentences are set aside as a test data set. Each of the sentences in the test set is used as an input sentence one by one, and the system returns its senses. The achieved results are presented in the following sub-section:

6.2.2. Train-Test Split for Evaluating Supervised Machine Learning Algorithms

In this section, the train-test split is used for any supervised ML models and it is used for clas sification problems. Table 6.1 Shows that discusses supervised machine learning models and hybrid model training and testing evaluation results. In this study, to evaluate the model perfo rmance, two mechanisms were used for training and testing the dataset. These are: train size =80% and test size =20%; and train size =75% and test size =25%. From these types of train_ test_split of the dataset, in this study, the best-performed performance to fit the model accura cy to predict correctly is train size = 80% and test size = 20% dataset split. So, in this training

and testing of the dataset, the best model to provide WSD for the Hadiyyisa language is the **N B-SVM-NN** model which provided an accuracy result of **0.909%**.

Table 6. 1 Supervised Machine learning Model and Hybrid Model Evaluation Result of Training and Testing Dataset

Models	NB	SVM	NN	NB-SVM	NB-NN	**SVM-NN**	**NB-SVM-NN**
Test size=20% Accuracy	0.86	0.88	0.87	0.874	0.857	0.88	**0.909**
Test size=25% Accuracy	0.848	0.878	0.85	0.874	0.848	0.878	**0.896**

Table 6.2 Shows discusses supervised machine learning models and hybrid ensemble models training and testing evaluation results. In this study, to evaluate the model performance here u sed two mechanisms to train_test_split of the dataset. These are: train size =80% and test size =20%; and train size =75% and test size =25%. From these types of train_test_split of the dat aset, in our study, train size =80% and test size =20% dataset split is the efficient train and tes t model accuracy to predict correctly. In this training and testing of the dataset, the best model to provide WSD for the Hadiyyisa language is **the DT-RF-GB** model. It provides a **0.860 %** accuracy result.

Table 6. 2 Ensemble Machine Learning Models Evaluation Result for Training and Testing Dataset

Models	DT	RF	GB	RF-GB	DT-RF	DT-GB	**DT-RF-GB**
Test size=20% Accuracy	0.831	0.853	0.815	0.850	0.857	0.850	**0.860**
Test size=25% Accuracy	0.805	0.855	0.802	0.841	0.846	0.844	0.856

Table 6.3 and Table 6.4 Shows the cross-validation result for supervised machine learning models and hybrid models. In this study, to validate the model performance, two cross-validation mechanisms were used for training and testing the dataset. These are: 5-fold Shuffle split Cross-validation and 10-fold Shuffle split cross-validation. In addition, 5-fold Shuffle split cross-validation has been found to have better validation results than 10-fold Shuffle split cross-validation techniques to validate the models.

Table 6. 3 Cross Validation Result of Supervised Machine learning Model and Hybrid Model

Models	NB	SVM	NN	NB-SVM	NB-NN	**SVM-NN**	**NB-SVM-NN**
5-fold Shuffle split	0.85	0.876	0.827	0.874	0.830	0.86	0.90
10-fold Shuffle split	0.844	0.86	0.827	0.869	0.832	0.856	0.893

Table 6. 4 Cross Validation Result of Supervised Machine Learning models and Ensemble Machine Learning Models

Models	DT	**RF**	GB	RF-GB	DT-RF	DT-GB	DT-RF-GB
5-fold Shuffle split	0.81	0.847	0.81	0.828	0.829	0.829	0.8445
10-fold Shuffle split	0.803	0.834	0.797	0.813	0.815	0.816	0.8318

6.2.3. Parameters Evaluation

Making predictions requires parameter evaluations. The final parameter evaluations found after training would decide how the model would be performed on unseen data. Table 6.5 shows the parameter evaluation for the SGD classifier. The Stochastic Gradient Descent Optimizer tries to find the minimum for a function. The function of interest, in this case, is the loss/error function. To minimize the error, there are use the SGD optimizer. The SGD optimizer works iteratively by moving in the direction of the gradient. The experimental result of the SGD classifier shows that the best hyperparameters are as follows: Parameter 3 has almost equal accuracy values of **0.88%**.

Table 6. 5 Hyperparameter Evaluations for Support Vector Machine Model

parameters	Loss	Penalty	Random_state	Alpha	max_iter	Tol	**SGD classifier accuracy**
Parameter 1	**Hinge**	**L2**	**42**	**0.0001**	**1000**	**1.e-3**	**0.88**
Parameter 2	Log	L2	42	0.001	1000	1.e-3	0.86
Parameter 3	squared_hinge	L2	42	0.001	1000	1.e-3	0.874

Table 6.6 shows the parameter evaluation for MLPclassifier and MultinomialNB. Which parameters are best for MLPclassifier and MultinomialNB? To answer this question from the exp

erimental result of the for MLPclassifier and MultinomialNB the best hyper parameters are li sted as follows: Alpha=0.0001, max_iter=1000, Hidden_layer= (6,), solver=lbfgs, activation=R elu or identity and Alpha=0.1 up to 0.4, Fit_prior=True, Class_prior=None are optimal accurac y value of **0.87** and **0.86** respectively.

Table 6. 6 Hyperparameters Evaluations for Neural Network and Naïve Bayes Models

Parameters	Alpha	max_iter	Hidden_layer	**Solver**	**activation**	**MLPClassifier Accuracy**
Default Parameter	0.0001	1000	(3,)	Lbfgs	**Relu or Identity**	0.86
Best Parameter	0.0001	1000	(6,)	Lbfgs	**Relu or identity**	0.87
Parameter1	0.0001	1000	(6,)	Lbfgs	Tanh or Logistic	0.85
Parameter2	0.0001	1000	(6,)	Adam	**Identity**	0.83
Parameter3	0.0001	1000	(6,)	Adam	**Relu**	0.86
Parameter4	0.0001	1000	(6,)	Adam	**Logistic**	0.866

Parameters	**Alpha**	**Fit_prior**	**Class_prior**	**MultinomialNB accuracy**
Parameter1	0.1	True	None	0.86
Paramter2	0.5	True	None	0.83
Parameter3	0.6	True	None	0.80
Parameter4	0.01	True	None	0.81

Table 6.7 shows the parameter evaluation for VotingClassifier. The experimental result shows that three classifiers append it as follows: MLPclassifier, MultinomialNB, and SGDClassifier. These three classifiers of hyperparameters are combined by the Voting Classifier, which provides an optimum accuracy value of **0.909 %**.

Table 6. 7 Hyperparameters Evaluations for Hybrid Model

Classifier	**Parameters**							
	Alpha	loss	penalty	hidden_ layer_sizes	tol	activ ation	solver	**Accurac y**
SGDClassifier	0.0001	hinge	L2		1.e-3			0.88
MultinomialNB	0.1							0.86
MLPClassifier	0.0001			(6,)		Relu	lbfgs	0.87
VotingClassifier	Estimators=[(‘ MultinomialNB’,NB), (‘ MLPClassifier’,NN), (‘SGDClassifier,SVM’),weights=[1,1,1]]							0.909

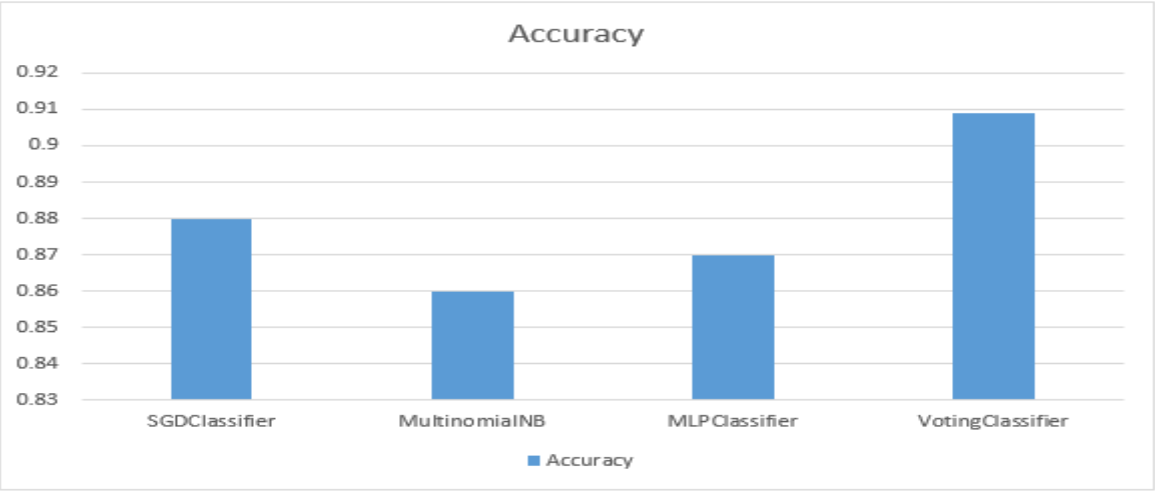


Figure 6. 1 Hyperparameter Result

Table 6.8 shows the parameter evaluation for supervised ensemble machine learning classifie rs. Which classifier is the best classifier to provide for WSD? From the experimental results o f the supervised ensemble machine learning classifiers, the Random ForestClassifier is the be st classifier. The Random ForestClassifier has included the following hyper parameters: Max _depth =20 or None, n_estimators =500 or 750, class_weight =balanced, optimum accuracy v alue of **0.8539**.

Table 6. 8 HyperParameters Tuning Evaluation for Three Supervised Ensemble Machine learning Models

Classifier							Accuracy
	Criterion	Min_samples_leaf	Min_samples_split	Max_depth	n_estimators	class_weight	
DecisionTree Classifier	Gini	1	3	None	-	-	**0.8313**
	entropy	1	2	15	-	-	0.812
	entropy	1	2	20	-	-	0.80
Random ForestClassifier		1	3	20	500, 750	Balanced	**0.8539**
		1	3	15	500	Balanced	0.832
		1	2 , 5	20, 15	500	balanced	0.84
GradientBoostingClassifier		1	2, 3	15	100	-	**0.8156**
		1	3	20	500	-	0.792
		1	2	None	500	-	0.781

Table 6.9 shows that the parameter evaluation for DecisionTreeClassifier, RandomForestClas sifier, and GradientBoostingClassifier. The experimental result shows that the appended three classifiers are as follows: DecisionTreeClassifier, RandomForestClassifier, and GradientBoos tingClassifier. These three classifiers' hyperparameters are appended by the Voting Classifier parameter of the **majority hard vote** to provide an optimum accuracy value of **0.86 .**

Table 6. 9 Parameters Evaluations for Hybrid Ensemble machine learning Model

Classifier	Parameters	Accuracy of classifier
Voting classifier	Estimators=[(‘ DecisionTreeClassifier’,DT), (‘RandomForestClassifier’,RF), (‘GradientBoostingClassifier, GB),weights=[1,1,1], voting='**hard**']	**0.86**
Voting classifier	Estimators=[(‘ DecisionTreeClassifier’,DT), (‘RandomForestClassifier’,RF), (‘GradientBoostingClassifier’, GB),weights=[1,1,1], voting= '**soft**']	0.84

6.2.4. Accuracy of Models

In this section, to find the accuracy of a model, to calculate the accuracy of a model by dividing the number of correctly predicted classes by the total number of samples. Furthermore, by using the sklearn accuracy_score() function, which takes in the true labels and predicted labels as arguments and displays the accuracy as a floating value. Therefore, this study mainly focused on the two selected models that have been generated with their accuracy values. These models are SVM and Hybrid models, which have been presented based on their model accuracy.

.

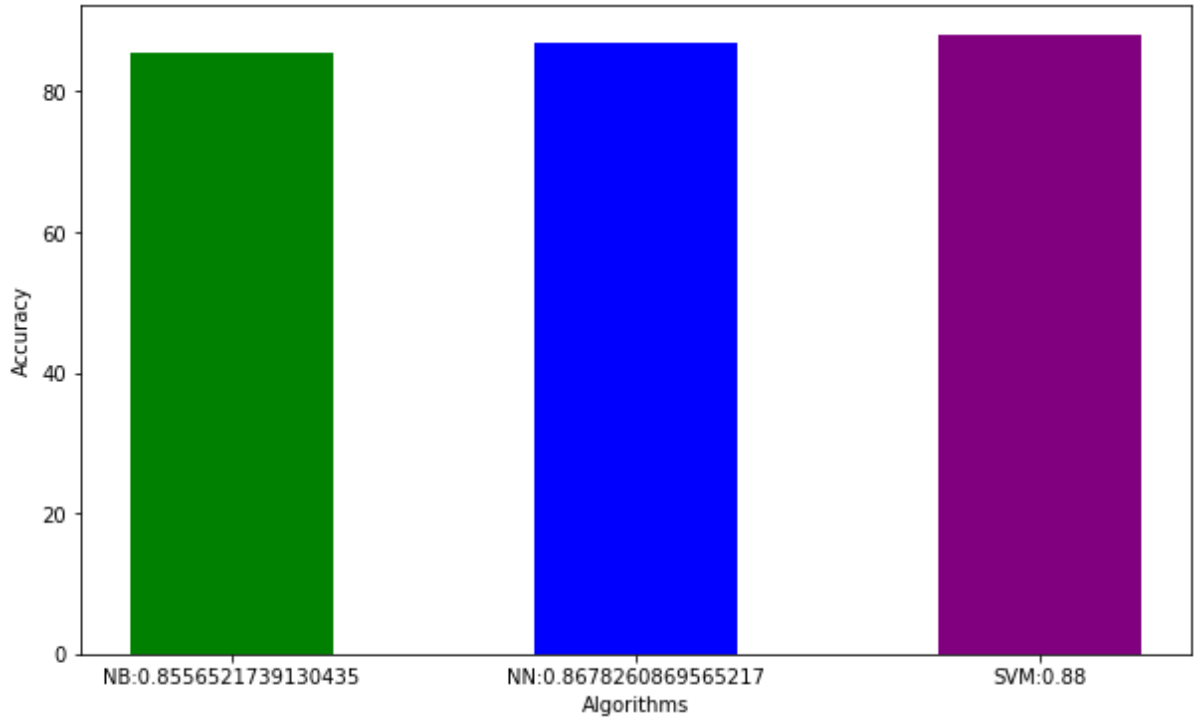


Figure 6. 2 Supervised Machine Learning Models Accuracy for Hadiyyisa Language

Figure 6.14 shows that from the supervised machine learning models for the ten ambiguous words, the SVM model has provided the best accuracy result. These models are appropriate and accurate to predict the correct class for these ten ambiguous words.

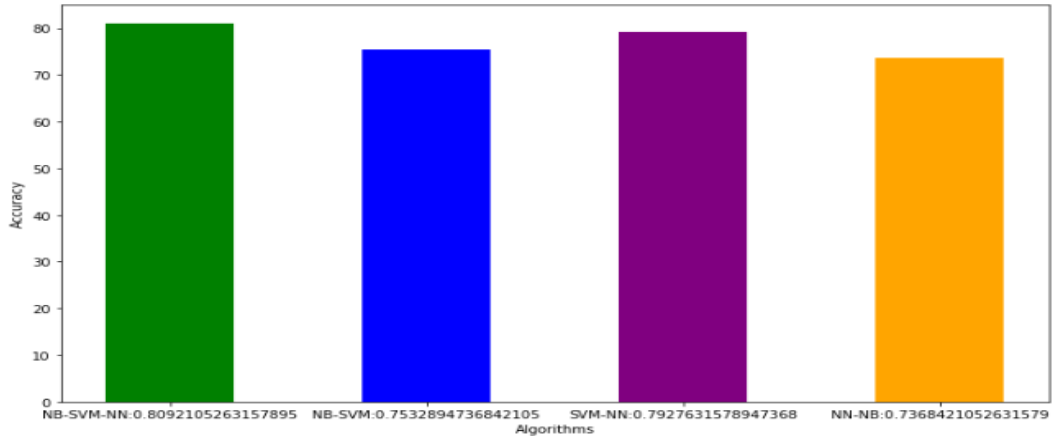


Figure 6. 3 Hybrid Models Accuracy for Hadiyyisa Language

Figure 6.15 shows that from the Hybrid models for the ten ambiguous words, the **NB-SVM-NN** model has provided the best accuracy result. These models are appropriate and accurate to predict the correct class for these ten ambiguous words.

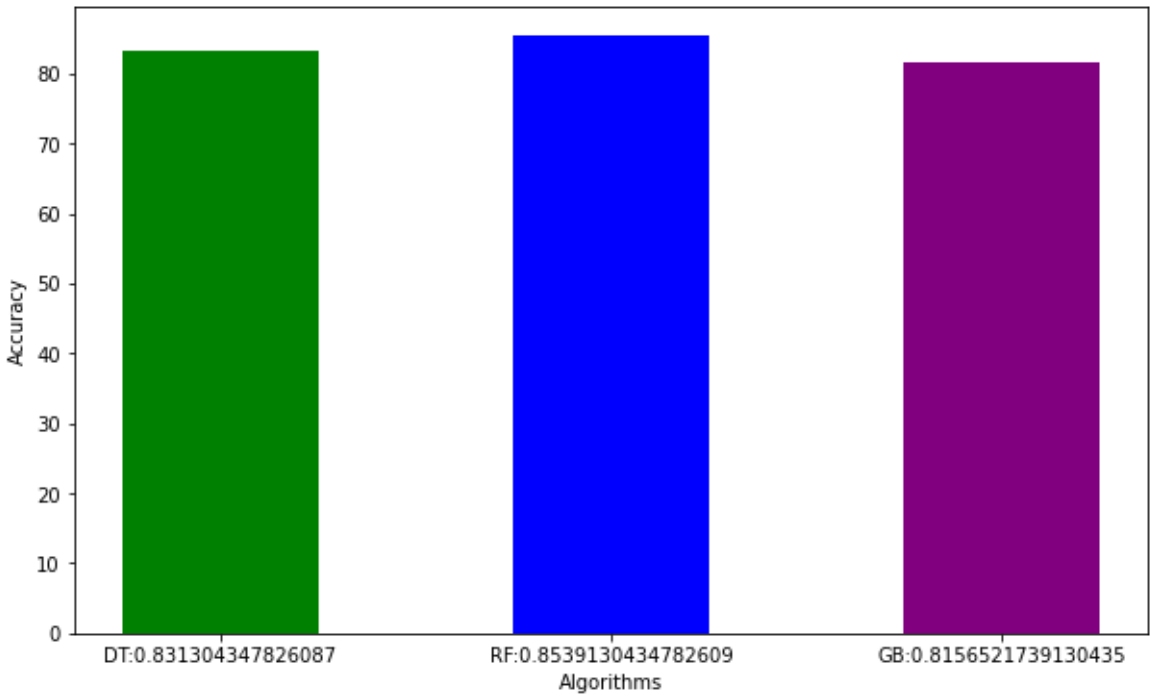


Figure 6. 4 Ensemble Supervised Machine Learning Models Accuracy

Figure 6.16 shows that from the supervised machine learning models for the ten ambiguous words, the RF model has provided the best accuracy result. These models are appropriate and accurate to predict the correct class for these ten ambiguous words.

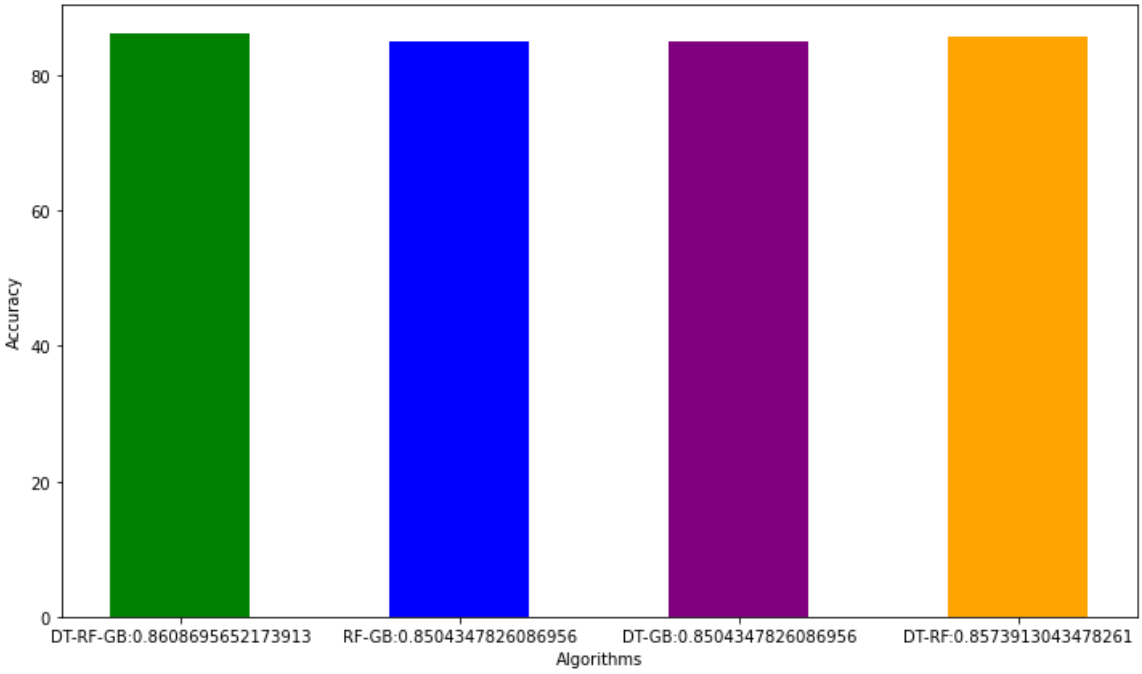
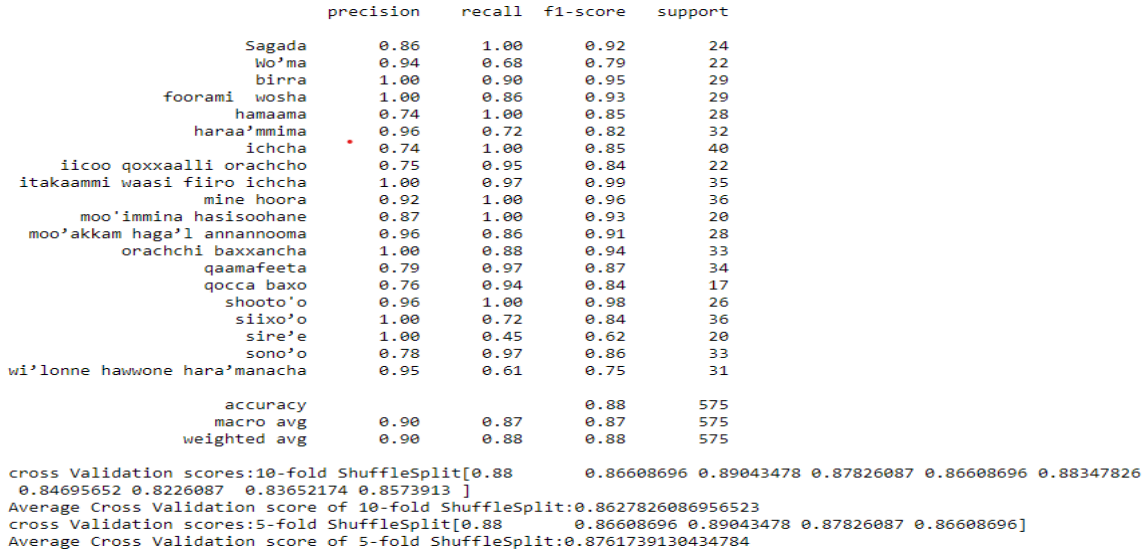


Figure 6. 5 Hybrid Ensemble Machine Learning Models Accuracy

Figure 6.17 shows that from the Hybrid ensemble machine learning models for the ten ambiguous words, the DT-RF-GB model has provided the best accuracy result. These models are appropriate and accurate to predict the correct class for these ten ambiguous words.

6.2.5. Classification Report and Confusion Matrix

This part discusses the classification report and confusion matrix for the above-compared accuracy result for the selected classifier, such as the **SVM**, NB, NN, and Hybrid (NB-SVM, NB-NN, SVM-NN, and **NB-SVM-NN**) Models. In addition, this study used two or more ensemble machine learning models, such as DT, RF, GB, and hybrid ensemble (DT-RF, DT-GB, RF-GB, and DT-RF-GB) models. From the above-listed models, this study chose four better models to handle WSD for the Hadiyyisa language: SVM, Hybrid model (NB-SVM-NN), RF, and Hybrid ensemble model (DT-RF-GB). Moreover, these four models are discussed based on the evaluation metrics and cross-validation methods. Monte Carlo Cross-Validation (Shuffle Split), which is appropriate to validate the actual accuracy of the models. Machine learning algorithms use classification reports as one of their performance evaluation indicators. It has been used to display our trained classification models' Precision, Recall, F1-Score, and Support. Precision means the ratio of true positives to the sum of true and false positives. Recall means the ratio of true positives to the sum of true positives and false negatives. The weighted harmonic mean of Precision and Recall is the F1-Score. The number of actual instances of the class in the dataset is referred to as support. The macro average between the classes is the average of Precision, Recall, and F1-Score. Between the classes, the weighted average of Precision, Recall, and F1-Score is calculated. Monte Carlo Cross-Validation (Shuffle Split) is an extremely bendable procedure of cross-validation. By using the classification report, accuracy results on the testing set, training set, and parameter tuning have been presented.



	precision	recall	f1-score	support
Sagada	0.86	1.00	0.92	24
Wo'ma	0.94	0.68	0.79	22
birra	1.00	0.90	0.95	29
foorami wosha	1.00	0.86	0.93	29
hamaama	0.74	1.00	0.85	28
haraa'mmima	0.96	0.72	0.82	32
ichcha	0.74	1.00	0.85	40
iicoo qoxxaalli orachcho	0.75	0.95	0.84	22
itakaammi waasi fiiro ichcha	1.00	0.97	0.99	35
mine hoora	0.92	1.00	0.96	36
moo'immina hasisoohane	0.87	1.00	0.93	20
moo'akkam haga'l annannooma	0.96	0.86	0.91	28
orachchi baxxancha	1.00	0.88	0.94	33
qaamafeeta	0.79	0.97	0.87	34
qocca baxo	0.76	0.94	0.84	17
shooto'o	0.96	1.00	0.98	26
siixo'o	1.00	0.72	0.84	36
sire'e	1.00	0.45	0.62	20
sono'o	0.78	0.97	0.86	33
wi'lonne hawwone hara'manacha	0.95	0.61	0.75	31
accuracy			0.88	575
macro avg	0.90	0.87	0.87	575
weighted avg	0.90	0.88	0.88	575

```
cross Validation scores:10-fold ShuffleSplit[0.88       0.86608696 0.89043478 0.87826087 0.86608696 0.88347826
 0.84695652 0.8226087  0.83652174 0.8573913 ]
Average Cross Validation score of 10-fold ShuffleSplit:0.8627826086956523
cross Validation scores:5-fold ShuffleSplit[0.88       0.86608696 0.89043478 0.87826087 0.86608696]
Average Cross Validation score of 5-fold ShuffleSplit:0.8761739130434784
```

Figure 6. 6 Support Vector Machine Classification Report

Figure 6.30 shows the confusion matrix and classification report for the ten target words. In general, 69 data points have been incorrectly classified while 506 data points have been correctly classified out of 575 tested data points. To validate model performance using the 10-fold Shuffle Split cross-validation test score.

	precision	recall	f1-score	support
Sagada	0.85	0.96	0.90	24
Wo’ma	0.79	0.68	0.73	22
birra	0.90	0.93	0.92	29
foorami wosha	0.96	0.90	0.93	29
hamaama	0.74	0.89	0.81	28
haraa’mmima	0.92	0.72	0.81	32
ichcha	0.71	0.97	0.82	40
iicoo qoxxaalli orachcho	0.72	0.82	0.77	22
itakaammi waasi fiiro ichcha	1.00	0.94	0.97	35
mine hoora	0.94	0.92	0.93	36
moo'immina hasisoohane	0.91	1.00	0.95	20
moo’akkam haga’l annannooma	0.88	0.82	0.85	28
orachchi baxxancha	0.97	0.85	0.90	33
qaamafeeta	0.78	0.91	0.84	34
qocca baxo	0.72	0.76	0.74	17
shooto'o	0.93	1.00	0.96	26
siixo’o	0.90	0.75	0.82	36
sire’e	0.92	0.55	0.69	20
sono’o	0.85	1.00	0.92	33
wi’lonne hawwone hara’manacha	1.00	0.68	0.81	31
accuracy			0.86	575
macro avg	0.87	0.85	0.85	575
weighted avg	0.87	0.86	0.86	575

```
cross Validation scores:ShuffleSplit[0.86086957 0.81913043 0.86956522 0.85913043 0.81391304 0.82956522
 0.83304348 0.78956522 0.81217391 0.83130435]
Average Cross Validation score of ShuffleSplit:0.8318260869565218
cross Validation scores:5-fold ShuffleSplit[0.86086957 0.81913043 0.86956522 0.85913043 0.81391304]
Average Cross Validation score of 5-fold ShuffleSplit:0.8445217391304347
```

Figure 6. 7 Ensemble Machine Learning Classification Report

```
Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0x26d7ca262e0>
```

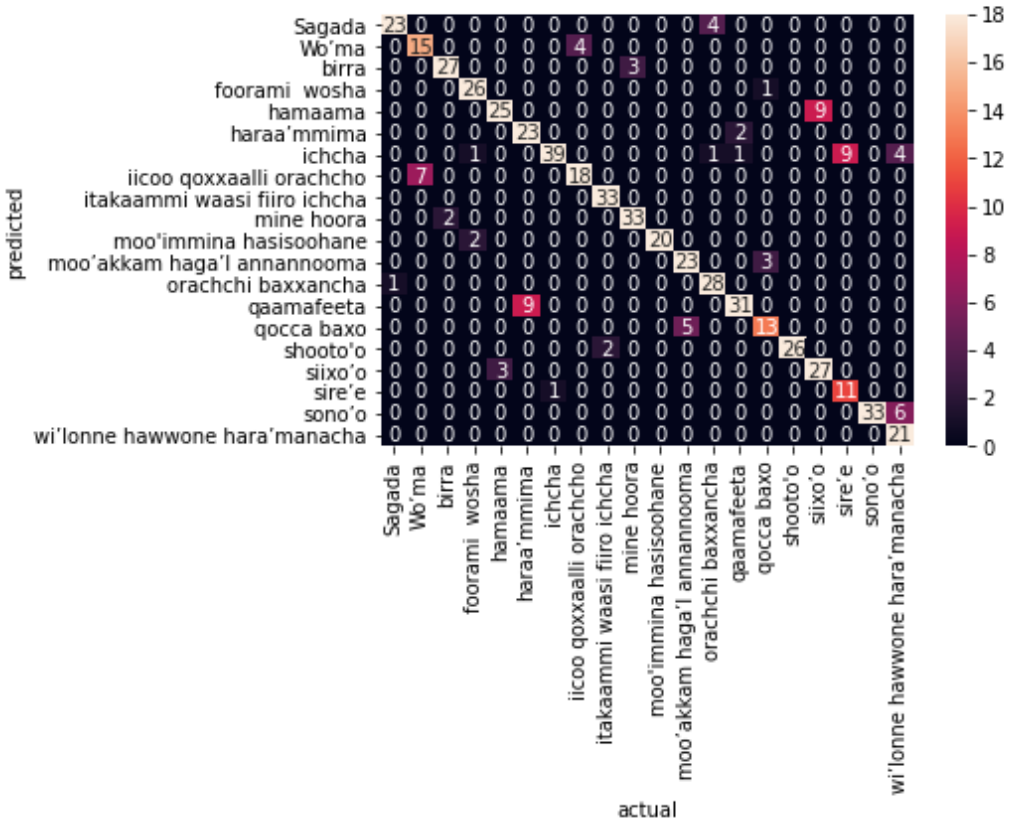


Figure 6. 8 Ensemble Machine Learning Classification matrix

Figure 6.32 shows the confusion matrix and classification report for the ten target words. In general, 86 data points have been incorrectly classified while 489 data points have been correctly classified out of 575 tested data points. To validate model performance using the 5-fold Shuffle Split cross-validation test score.

	precision	recall	f1-score	support
Sagada	0.86	1.00	0.92	24
Wo’ma	0.94	0.73	0.82	22
birra	1.00	0.93	0.96	29
foorami wosha	0.96	0.86	0.91	29
hamaama	0.88	0.82	0.85	28
haraa’mmima	0.89	0.97	0.93	32
ichcha	0.86	0.93	0.89	40
iicoo qoxxaalli orachcho	0.78	0.95	0.86	22
itakaammi waasi fiiro ichcha	1.00	0.97	0.99	35
mine hoora	0.97	1.00	0.99	36
moo’immina hasisoohane	0.87	1.00	0.93	20
moo’akkam haga’l annannooma	0.96	0.89	0.93	28
orachchi baxxancha	1.00	0.88	0.94	33
qaamafeeta	0.94	0.94	0.94	34
qocca baxo	0.78	0.82	0.80	17
shooto’o	0.93	1.00	0.96	26
siixo’o	0.86	0.86	0.86	36
sire’e	0.84	0.80	0.82	20
sono’o	0.86	0.97	0.91	33
wi’lonne hawwone hara’manacha	0.96	0.77	0.86	31
accuracy			0.91	575
macro avg	0.91	0.91	0.90	575
weighted avg	0.91	0.91	0.91	575

```
cross Validation scores:10-fold ShuffleSplit[0.90956522 0.88347826 0.90434783 0.89043478 0.91478261 0.89913043
 0.87304348 0.86782609 0.88347826 0.90782609]
Average Cross Validation score of 10-fold ShuffleSplit:0.8933913043478261
cross Validation scores:5-fold ShuffleSplit[0.90956522 0.88347826 0.90434783 0.89043478 0.91478261]
Average Cross Validation score of 5-fold ShuffleSplit:0.9005217391304348
```

Figure 6. 9 Hybrid Model Classification Report for Multi-Class

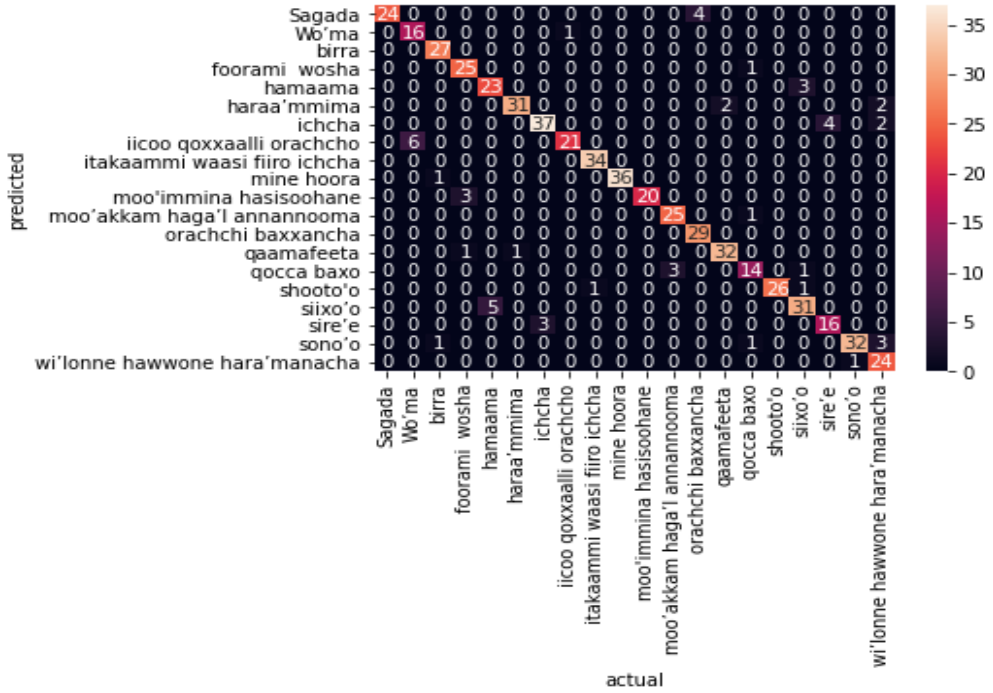


Figure 6. 10 Hybrid Model Classification matrix

Figure 6.33 shows the confusion matrix and classification report for the ten target words. In general, 52 data points have been incorrectly classified while 523 data points have been correctly classified out of 575 tested data points. To validate model performance using the 5-fold Shuffle Split cross-validation test score.

6.2.6. Summary of the Result and Discussion

In this study, four experiments were conducted using a supervised machine learning classifier and a Hybrid ensemble machine learning classifier from the Skit-learn library. They also use manually sense-annotated datasets, labeled datasets, and classification algorithms to complete classification jobs. In this study, six trials were carried out utilizing six different classification algorithms. These classification algorithms are **Naive Bayes, Support Vector Machine, Neu ral Network, Decision Tree, Random Forest, Gradient Boosting,** and **Hybrid Model**. In a ddition, validating a model using the appropriate cross-validation like the Monte Carlo (Shuff le Split) method was verified. In addition, these classification algorithms should be applied to the ten selected ambiguous words in the Hadiyyisa language. These ambiguous words are An ga, Diinate, Hagara, Hurbaata, Misha,Bu'o, Inqe, biqe, Ille and Seera. The total dataset dimen sion contains 2872

sentences for six ambiguous words. From the total dataset, split a dataset into the training set and testing set, there are 2297 and 575sentences, respectively. Finally, this study has answere d the following questions with experimental evidence:

- Which algorithm provides better performance for Hadiyyisa language WSD from the supervised machine learning models?
- Which model provides the best performance for Hadiyyisa language WSD from the Hybrid models?

Experiment I: - Which Algorithm provides better performance for Hadiyyisa language WSD from the supervised machine learning models?

To discuss this experiment depending on the above Section 6.1.3 Accuracy of model experimentation results, using the Python programming language and Jupyter Notebook to use necessary NLP research development tools. To do the experimentation for each selected supervised machine learning model to compare the better result. To show the best model with accuracy and with their average cross-validation score have been shown below the table. Table 6.10 shows that from those four classifier algorithms,SVM models and **SVM-NN-NB** are the best result of accuracy to predict actual sense for Hadiyyisa language.

Table 6. 10 Evaluation metrics

Accuracy score	Precision score	Recall score	f1_score	Models
0.855652173913	0.8626184640	0.8556521739	0.85445908593	NB
0.88	0.9009324846	0.88	0.87624776987	**SVM**
0.867826086956	0.8735911100	0.8678260869	0.86773122890	NN
0.909565217391	0.9141788929	0.9095652173	0.90905727696	**SVM-NN-NB**
0.874782608695	0.8830806102	0.8747826086	0.87439493493	NB-SVM
0.883478260869	0.8927924561	0.8834782608	0.88236725961	SVM-NN
0.874782608695	0.8830806102	0.8747826086	0.87439493493	NN-NB

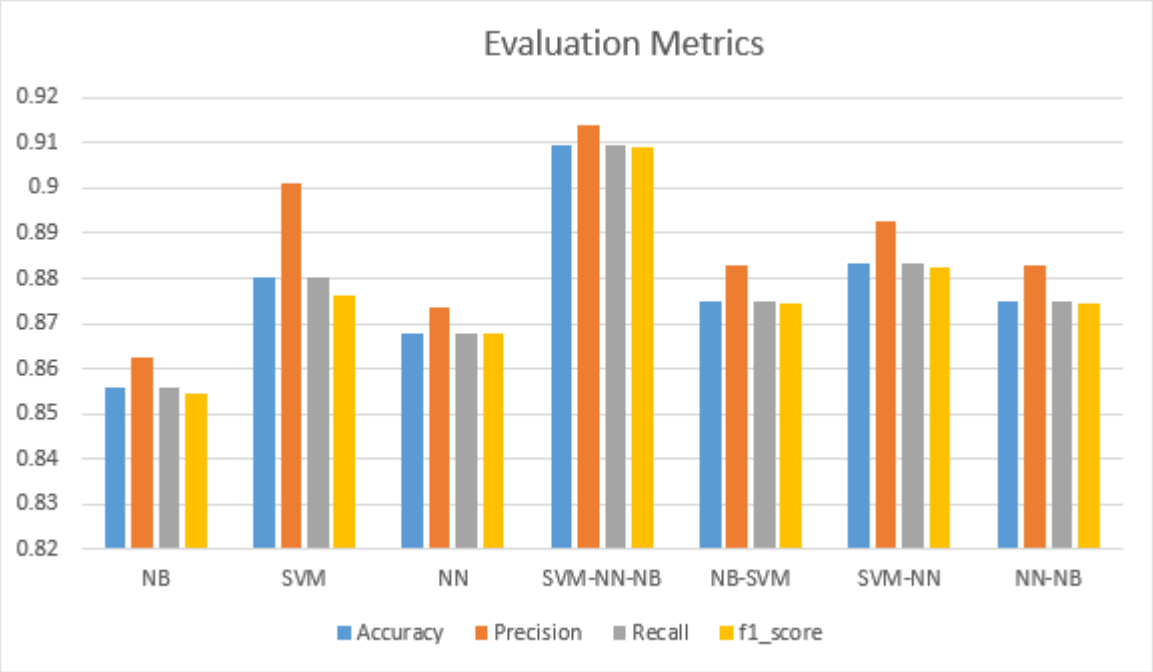


Figure 6. 11 Overall Model Evaluation Metrics

Experiment III: - Which model provides the best performance for Hadiyyisa language WSD from the Hybrid models?

The ability to answer this question was dependening on the experimental evaluation results and the comparison of hybrid model results. The best Hybrid models for the Hadiyyisa language are briefly summarized as follows in Table 6.13 Below. Table 6.13 shows that **NB-SVM-NN** is the best Hybrid model to provide the accuracy of WSD for the Hadiyyisa language to predict the correct sense of meaning.

Table 6. 11 Comparison of Hybrid Models with Cross-validation result

Cross-validation	**Hybrid models**			
	NB-SVM-NN	NB-SVM	SVM-NN	NB-NN
5- Fold Shuffle split cross-validation	0.9005	0.874	0.856	0.83

6.2.7. Comparison of the Existing Study of the Researchers

This section discussed the existing study of the researchers, which is related to this study. It is to compare and contrast this study with the existing related work, which is shown in Table 6.14. Table 6.14 shows the results, methodology, and language of study areas for the other related existing researchers discussed below.

Table 6. 12 Comparison of the Existing Study of the Researchers

Sno	Authors	Methodology	Result	Languages
1.	(Tesema, 2016)	Using a Hybrid approach(unsupervised machine learning and Rule-based approaches)	The unsupervised strategy had a system accuracy of 76.05 percent, while the hybrid approach had an accuracy of 89.47 percent.	Afaan Oromo
2.	Teklay Muruts Welemichael (2018)	He uses a semi-supervised machine learning method	The ADTree, SMO, AdaBoostM1, Bagging, and Naive Bayes algorithms were used to achieve 93.6636 percent, 91.9224 percent, 85.35 percent, 79.1917 percent, and 70.8968 percent, respectively.	Tigrigna Language
3.	Temesgen Tadesse Tilahun(2021)	He uses Machine Learning Algorithms	On WS11 (5-5) using 10-fold cross-validation, four algorithms, Support Vector Classifier and Bagging Classifiers with TF-IDF, respectively, achieved an accuracy of 83.22 percent and 82.82 percent.	Wolaita language
4.	Amlakie Aschale and Kinde Anlay Fante(2021)	They used Corpus-based techniques	According to the findings of the studies (semi-supervised machine learning approach), the ADtree algorithm produces the best results. The proposed method achieved an average performance of 92.1 percent, 91.3 percent, 91 percent, and 91.1 percent in Precision, Recall, F1-score, and Accuracy, respectively, using the ADTree algorithm.	Ge'ez Language

5.	My thesis works deals	using supervised ML models	To validate the proposed models, such as 5-fold Shuffle Split Cross-Validation was used. The proposed method achieved an average performance of **0.909%, 0.914%, 0.909 %,** and **0.909 %** in Accuracy , Precision ,Recall, and F1-score, respectively, using Hybrid**(SVM-NN-NB)** model.	Hadiyyisa language

6.2.8. Contribution of the Study

The main contributions of this study to word sense disambiguation in the Hadiyyisa language are presented as follows:

1. Create the required data-preprocessing phase for the Hadiyyisa Language. Stop word removal, lower letter, punctuation removal, number removal, and tokenization are used to improve data quality and reduce data sparseness.
2. Design WSD dataset preparation.
3. Customize the WSD Model for Hadiyyisa.

The contribution of this thesis is to investigate the WSD model for the Hadiyyisa language using supervised machine learning methods and integrating supervised ML models. Hadiyyisa is a commonly spoken language with a scarcity of natural language corpora. As stated in the above section, my contribution is a newly established and accessible benchmark dataset for the Hadiyyisa lexical sample WSD challenge. This newly established lexical sample training dataset has important for the next researcher in the Hadiyyisa language.

CONCLUSION, RECOMMENDATION

7.1. Conclusion

This thesis is dedicated to the problem of word-sense disambiguation in natural language processing. The WSD problem is to determine what sense corresponds to an ambiguous word depending on the context. In this study, relations between a word's sense and its surrounding context are examined.

In this thesis, four experiments have been conducted using different supervised machine learning algorithms. The first experiment was a comparison of the results obtained using the three supervised machine learning methods which are SVM, NB, and NN models. In this experiment, the proposed SVM model has performed better results than the other NB and NN models. The second experiment was a comparison of the results obtained using the four Hybrid models which are SVM-NB, NB-NN, SVM-NN, and SVM-NB-NN models. In this experiment, the proposed SVM-NB-NN model has performed better results than the other Hybrid model. The third experiment was a comparison of the results obtained using the parameter evaluation mechanisms for each supervised machine learning model. The fourth experiment shows evaluating the training and testing dataset split. In this study, to evaluate the model performance, two mechanisms were used for training and testing the dataset. These are: train size =80% and test size =20%; and train size =75% and test size =25%. In this experiment, a training size =80% and testing size =20% performed better than the other training size =75% and testing size =25%.

As a result, finding the correct senses (meanings) of polysemy words in context also improves the NLP applications' efficiency effectively. In this study, the WSD of the Hadiyyisa language was investigated to improve the performance of NLP applications undertaken by the next researchers who will conduct NLP-related studies. Text retrieval system, text summarization, speech processing, text processing, POS, automatic sentence parser, spell checker, and other NLP applications require Hadiyyisa WSD. The lack of WSD in Hadiyyisa limits natural language processing systems and WSD to solving computers' understanding of Hadiyyisa and constructing datasets for computers to learn or understand the proper sense of ambiguous words with correct contextual meanings.

This study was implemented using supervised machine learning models and hybrid models.

Supervised machine learning methods such as Naive Bayes, Support Vector Machines, Neura l Networks, Decision Tree, Random Forest, Gradient Boosting, and Hybrid models were impl emented. The proposed Hybrid(SVM-NN-NB) model achieved an average performance of **0. 909%, 0.914%, 0.909 %,** and **0.909 %** in Accuracy , Precision ,Recall, and F1-score respect ively. In particular, for classifying the correct senses of classes, the proposed **Hybrid** model i s found to have more accurate results than other models.

7.2. Recommendations

This thesis also serves as a review of Hadiyyisa Language WSD work, as well as its challenges and issues. Lack of linguistic resources, such as the thesaurus, WordNet, machine-readable dictionaries, and effective Hadiyyisa language resources, which are needed for this research. We should produce datasets based on the study's functionality as the first researcher, as these will be highly beneficial to subsequent researchers who will use them to grow the data size and update the number of ambiguous words. As a result, this study attempted to fill a gap in the Hadiyyisa language. Word sense disambiguation is still a research area in every language. Another problem was the lack of sense-annotated data for the language, which forced me to limit our study to ten ambiguous words in the language. In addition, the total data size to train and test is limited to 2872 sentences due to a lack of meaningful annotated data. Therefore, the following recommendations which include the development of resources and future research directions for WSD of Hadiyyisa language:-

- The next researcher employing a supervised machine learning method for more than ten ambiguous terms in this study will be advisable as increasing the dataset size has been regarded to be more realistic for the WSD model.
- The result of WSD tests shows the TF-IDF method produces encouraging results. However, the next researcher will be doing other feature extraction techniques (word2vec).
- Therefore, a Hybrid (SVM-NB-NN) model is recommended to effectively predict the most appropriate interpretation of an ambiguous word. In addition to that, this study contributes to developing and evaluating supervised machine-learning models to predict the correct senses of classes, it is better to extend to a deep learning approach model by incorporating a large dataset.

==========

==========

SPECIAL ACKNOWLEDGMENT

This research work was funded by Adama Science and Technology University under grant number: ASTU/SM-R/491/22 Adama, Ethiopia

REFERENCES

Abid, M., Habib, A., Ashraf, J., and Shahid, A. (2018). Urdu Word Sense Disambiguation Using Machine Learning Approach. Cluster Computing, 21(1), 515–522. https://doi.org/10.1007/s10586-017-0918-0

Alemu, A. A., and Fante, K. A. (2021). Corpus-Based Word Sense Disambiguation for Ge ' ez Language Corpus-Based Word Sense Disambiguation for Ge ' ez Language. January. https://doi.org/10.20372/ejssdastu

Assemu, S. (2011). Unsupervised Machine Learning Approach for Word Sense Disambiguation to Amharic Words.

Bade, G. Y. (2021). Natural Language Processing and Its Challenges on Omotic Language Group of Ethiopia. 03(04), 26–30.

Bala, P. (2013). Knowledge-Based Approach for Word Sense Disambiguation using Hindi Wordnet. 36–41.

Bevilacqua, M., Pasini, T., Raganato, A., and Navigli, R. (2021). Recent Trends in Word Sense Disambiguation: A Survey. 4330–4338.

Bhadane, C., Thoutam, N., Mahajan, M., and Suryawanshi, V. (2021). Word Sense Disambiguation (WSD) using Neural Networks. 3, 1831–1838.

Calvo, H., Rocha-Ramirez, A. P., Moreno-Armendariz, M. A., and Duchanoy, C. A. (2019). Toward Universal, Word Sense Disambiguation Using Deep Neural Networks. IEEE Access, 7, 60264–60275. https://doi.org/10.1109/ACCESS.2019.2914921

Chaplot, D. S., and Salakhutdinov, R. (2018). Knowledge-based word sense disambiguation using topic models. 32nd AAAI Conference on Artificial Intelligence, AAAI 2018, 5062–5069.

Chen, L. (2019). Word Sense Disambiguation Using Wikipedia Link Graph. 2019 IEEE International Conference on Big Data (Big Data), 6235–6236.

Dawn, D. Das. (2020). A comprehensive review of Bengali word sense disambiguation. Artificial Intelligence Review, 53(6), 4183–4213. https://doi.org/10.1007/s10462-019-09790-9

Demidova, L. A., Klyueva, I. A., and Pylkin, A. N. (2019). Hybrid Approach to Improving the Results of the SVM Hybrid Approach to Improving the Results of the SVM Classification Using the Random Forest Algorithm. Procedia Computer Science, 150, 455–461. https://doi.org/10.1016/j.procs.2019.02.077

Demilie, W. B. (2020). Multilingual Spelling Checker for Selected Ethiopian Languages. International Journal of Advanced Science and Technology, 29(7), 2641–2648.

Dereje, Y. (2017). Supervised automatic sentence parsing for Hadiya language.

Eluri, S., and Pilli, V. K. (2020). Global Word Sense Disambiguation of Polysemous Words in the Telugu Language. International Journal of Engineering and Advanced Technology, 10(1), 420–425. https://doi.org/10.35940/ijeat.a1915.1010120

Eneko Agirre, O. L. de L. (2010). Random Walks for Knowledge-Based Word Sense Disambiguation. Dissertation Abstracts International, B: Sciences and Engineering, 70(8), 4943. https://doi.org/10.1162/COLI

Escudero, G., Márquez, L., and Rigau, G. (2000). Boosting applied to word sense disambiguation. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 1810(August), 129–141. https://doi.org/10.1007/3-540-45164-1_14

Eshetu, A., Teshome, G., and Abebe, T. (2020). Learning Word and Sub-word Vectors for Amharic (Less Resourced Language). International Journal of Advanced Engineering Research and Science, 7(8), 358–366. https://doi.org/10.22161/ijaers.78.39

Faisal, E. (2018). Word Sense Disambiguation in Bahasa Indonesia Using SVM. 2018 International Seminar on Application for Technology of Information and Communication, 239–243.

Garkebo, T. S. (2015). Documentation and Description of Hadiya.

Godisso, S. H. (2017). Contrastive Analysis of Lexical Standardization in Amharic and Hadiya. January.

Gonfa, S. (2018). Word Sense Disambiguation for Afaan Oromo: Using Knowledge Base. August.

Gottschalk, K., Graham, S., Kreger, H., and Snell, J. (2002). Introduction to Web Clustering. 41(2).

Gupta, P., Dr.AnuragAwasthi, and RiteshRastogi. (2013). Impact of Word Sense Ambiguity for English Language in Web IR. International Journal of Engineering Research and Technology, 2(2), 1–8.

Gupta, S., Namavari, A., and Smith, T. O. (2017). Word Sense Disambiguation Using Skip-Gram and LSTM Models. 1–9.

Han, S., and Shirai, K. (2021). Unsupervised word sense disambiguation based on word embedding and collocation. ICAART 2021 - Proceedings of the 13th International

Conference on Agents and Artificial Intelligence, 2(Icaart), 1218–1225. https://doi.org/10.5220/0010380112181225

Hassen, S. (2015). Amharic Word Sense Disambiguation Using Wordnet. March.

Heo, Y., Kang, S., and Seo, J. (2020). Hybrid Sense Classification Method for Large-Scale Word Sense Disambiguation. IEEE Access, 8, 27247–27256. https://doi.org/10.1109/ACCESS.2020.2970436

Iacobacci, I., Pilehvar, M. T., and Navigli, R. (2016). Embeddings for word sense disambiguation: An evaluation study. 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 - Long Papers, 2(2003), 897–907. https://doi.org/10.18653/v1/p16-1085

Kacha, D. (2019). Part of Speech Tagger for Hadiyyisa Language.

Kang, M. Y., Min, T. H., and Lee, J. S. (2018). Sense Space for Word Sense Disambiguation. Proceedings - 2018 IEEE International Conference on Big Data and Smart Computing, BigComp 2018, 1, 669–672. https://doi.org/10.1109/BigComp.2018.00122

Karwa, R., and Chandak, and M. (2015). An Hybrid Approach To Word Sense Disambiguation With And Without Learned Knowledge. 4(2), 31–42.

Kebede, T. (2013). Word Sense Disambiguation for Afaan Oromo Language.

Kilgarriff, A. (1998). Gold standard datasets for evaluating word sense disambiguation programs. Computer Speech and Language, 12(4), 453–472. https://doi.org/10.1006/csla.1998.0108

Kim, M., and Kwon, H. C. (2021). Word Sense Disambiguation Using Prior Probability Estimation Based on the Korean WordNet.

Li, W., and Suzuki, E. (2020). Hybrid context-aware word sense disambiguation in topic modeling based document representation. Proceedings - IEEE International Conference on Data Mining, ICDM, 2020-Novem(Icdm), 332–341. https://doi.org/10.1109/ICDM50108.2020.00042

Loureiro, D., Tec, L. I., and Camacho-collados, J. (2021). Analysis and Evaluation of Language Models for Word Sense Disambiguation. February.

Melamud, O., Goldberger, J., and Dagan, I. (2016). context2vec: Learning generic context embedding with bidirectional LSTM. CoNLL 2016 - 20th SIGNLL Conference on Computational Natural Language Learning, Proceedings, 51–61. https://doi.org/10.18653/v1/k16-1006

Mikael, K., and Salomonsson, H. (2016). Word Sense Disambiguation using a Bidirectional

LSTM. June.

Miškovic, V. (2014). Machine Learning of Hybrid Classification Models for Decision Support. 318–323. https://doi.org/10.15308/sinteza-2014-318-323

Montoyo, A., and Su, A. (2005). Combining Knowledge- and Corpus-based Word-Sense-Disambiguation Methods. 23, 299–330.

Mukti Desai, M. K. B. (2013). Word sense disambiguation. Handbook of Natural Language Processing, Second Edition, 2(10), 315–338. https://doi.org/10.4249/scholarpedia.4358

Muruts, T. (2018). Word Sense Disambiguation for Tigrigna Language Using Semi-Supervised Machine Learning Approach.

Nancy Ide, J. V. (2004). Introduction to the special issue on word sense disambiguation. Computer Speech and Language, 18(3 SPEC. ISS.), 201–207. https://doi.org/10.1016/j.csl.2004.05.005

Navigli, R. (2009). Word Sense Disambiguation : A Survey. 41(2). https://doi.org/10.1145/1459352.1459355

Nithyanandan, S., and C, R. (2019). Deep Learning Models For Word Sense Disambiguation: A Comparative Study. SSRN Electronic Journal. https://doi.org/10.2139/ssrn.3437615

Pal, A. R., Munshi, A., and Saha, and D. (2013). An Approach to Speed-Up the Word Sense Disambiguation Procedure through Sense Filtering. 3(4), 29–41.

Pal, A. R., and Saha, D. (2015). Word Sense Disambiguation: A Survey. 5(3), 1–16.

Papandrea, S., Raganato, A., and Delli Bovi, C. (2017). SUPWSD: A flexible toolkit for supervised word sense disambiguation. EMNLP 2017 - Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Proceedings, 103–108. https://doi.org/10.18653/v1/d17-2018

Patil, V. H., Dey, N., and Mahalle, P. N. (2020). Lecture Notes in Networks and Systems 169 Proceeding of First Doctoral Symposium on Natural Computing Research.

Popov, A. (2018). Neural Network Models for Word Sense Disambiguation : An Overview. 18(1), 139–151. https://doi.org/10.2478/cait-2018-0012

Rahman, N., and Borah, B. (2021). An unsupervised method for word sense disambiguation. Journal of King Saud University - Computer and Information Sciences, XXXX. https://doi.org/10.1016/j.jksuci.2021.07.022

Raulji, J. K. (2016). Stop-Word Removal Algorithm and its Implementation for Sanskrit Language. 150(2), 15–17.

Salt, L., Jurdak, R., and Oliver, E. (2016). Hybrid Ensemble Learning for Triggering of GPS

in Long-Term Tracking Applications. September.

Sharma, N., and Niranjan, P. S. (2015). Applications of Word Sense Disambiguation : A Historical Perspective. 3(10), 1–4.

Singh, V. P., and Kumar, P. (2019). Sense disambiguation for Punjabi language using supervised machine learning techniques. Sadhana - Academy Proceedings in Engineering Sciences, 44(11). https://doi.org/10.1007/s12046-019-1206-x

Solomon, M. (2010). Word Sense Disambiguation for Amharic Text: a Machine Learning Approach. Unpublished Master's Thesis Addis Ababa University., 1, 10–105.

Sreenivasan, D., and Vidya, M. (2016). A Walk Through the Approaches of Word Sense Disambiguation. International Journal for Innovative Research in …, 2(10), 218–224. http://www.ijirst.org/articles/IJIRSTV2I10025.pdf

Tadesse, T. (2021). Word Sense Disambiguation for Wolaita Language Using Machine Learning Approach. August.

Taghipour, K., and Ng, H. T. (2015). One million sense-tagged instances for word sense disambiguation and induction. CoNLL 2015 - 19th Conference on Computational Natural Language Learning, Proceedings, 338–344. https://doi.org/10.18653/v1/k15-1037

Tesema, W., Tesfaye, D., and Kibebew, T. (2017). Towards the Sense Disambiguation of Afan Oromo Words Using Hybrid Approach (Unsupervised Machine Learning and Rule-Based). 61–77.

Tesema, Y. (2016). Hybrid Word Sense Disambiguation Approach for Afaan Oromo Words (Issue June).

Tesfaye, B. (2017). Afaan Oromo Word Sense Disambiguation Using WordNet.

Thankappan, R., Kala, M. T., and Joseph, D. (2016). Word Sense Disambiguation Using Decision Tree Method. 1(5), 1–6.

Vial, L., Lecouteux, B., and Schwab, D. (2020). Sense vocabulary compression through the semantic knowledge of wordNet for neural word sense disambiguation. Proceedings of the 10th Global WordNet Conference, 108–117.

Wahle, J. P., Ruas, T., Meuschke, N., and Gipp, B. (2021). Incorporating Word Sense Disambiguation in Neural Language Models. http://arxiv.org/abs/2106.07967

Woldeyohanes, D. (2016). Design and Development of Hadiyyisa Text Retrieval.

Xue-Ren, Shao-he Lv, X.-D. (2017). Chinese word sense disambiguation using a LSTM. 01027, 1–5.

Zeynep, A. (2010). Word Sense Disambiguation: Methods and Applications. 1–10.

Zhang, C. X., Liu, R., Gao, X. Y., and Yu, B. (2021). Graph Convolutional Network for Word Sense Disambiguation. Discrete Dynamics in Nature and Society, 2021. https://doi.org/10.1155/2021/2822126

APPENDICES

Annex A: Orthographic Representation of the Hadiyyisa

Phonemes in IPA	Upper case graphemes	Lower case graphemes	Ethiopic script
/a/	A	A	አ
/b/	B	B	ባ
/ tʃ ’/	C	C	ጭ
/d/	D	d	ዳ
/e/	E	e	ኤ
/f/	F	F	ፋ
/g/	G	G	ጋ
/h/	H	H	ሃ
/i/	I	I	ኢ
/dʒ/	J	J	ጃ
/k/	K	K	ካ
/l/	L	L	ላ
/m/	M	M	ማ
/n/	N	N	ና
/o/	O	O	ኦ
/k’/	Q	Q	ቃ
/r/	R	R	ራ
/s/	S	S	ሳ
/t/	T	T	ታ
/u/	U	U	ኡ
/w/	W	W	ዋ
/t’/	X	X	ጣ
/y/	Y	Y	ያ
/z/	Z	Z	ዛ

Annex C Stop Word Lists

Stop words
Eebikkina
Ixxi
Ise
Ka
te’im
Eese
Neesee
Nees
Neesdu
Neesem
Ihukkaarem
Isee
Bikkinaa
Ee
Ee’isam
Ni
Ihukkaarem
Ixxuwwa
Eebikkinam
Ki’nuwwi
Eekkam
Ee lasage
Ka bikkina
ki’nuwwi
Ihukkisa

Stop words
Odim
Isse
An
Lasage
Beelas
ihuta’nim
Ki
Ee’isam
ihuta’n
Er
Eebikkina
I
kobi’lishshina
m.k
At
Mahina
Mashka’im
ihu bikkina
Eeballi
Iyyummi
Issi
Ku
ee illage
Mato
Lamo

Annex D Sample Code for Train, Test and predict sentence and WSD for Hadiyyisa text

```
In [39]: X_train.values

Out[39]: array(['siidukki hurbaatame qeeraa  l allabinne hixixaa  aa aq woronne aagisookko ',
       'anga oobba', 'usheexxaxxi losaniins egeramo misha', ...,
       'gaga xanubee  xumammoc xumammoc hagara ', 'xaara tumaa  n misha',
       'lelli seera lonsiminne hindiyyaaxxi gaagaayyaato fuu  lisamo'],
      dtype=object)
```

```
In [20]: print(X_test[:10])

51       raashi ulli itophei  na qaxanchi muutina anga ...
168                          xulbe  i issi anga usheexxoo
1469           asheebi caata gameukka seera biira qaaxakko
926                  afeejjaarine waaraa hurbaata murokko
422      diinate ha  n orachchi haalattinne galchchiinn...
1090     anceete tommi baxxanchi hawweenanne hig misha ...
585                        yakite issoo xum sawwixxi hagara
220                                          anga do  isimma
1468      waru koyi seera hurbaata horaakoha dumma marukko
1440                               ludagi seera daanamukko
Name: sentence, dtype: object
```

```
In [21]: print(y_test[:10])

51                           haraa'mmima
168                           qaamafeeta
1469     wi'lonne hawwone hara'manacha
926                             sire'e
422                           mine hoora
1090                             siixo'o
585                           qocca baxo
220                           qaamafeeta
1468     wi'lonne hawwone hara'manacha
1440     wi'lonne hawwone hara'manacha
Name: label, dtype: object
```

```
In [22]: print(y_pred[:10])

['haraa'mmima' 'qaamafeeta' 'wi'lonne hawwone hara'manacha' 'sire'e'
 'birra' 'siixo'o' 'qocca baxo' 'qaamafeeta' 'sire'e' 'sono'o']
```

Annex E Confirmation of Data collection and Data Annotation