



**Ethiopian Language Identification from Text Data  
Using Hybrid Approach**

By

**SABA SITOTAW AMARE**

To

**The Faculty of Informatics of St. Mary's University**

**In Partial Fulfillment of the Requirements for the  
Degree of Master of Science**

**In  
Computer Science**

February, 2024

ACCEPTANCE

**Ethiopian Language Identification from Text Data  
Using Hybrid Approach**

By

**SABA SITOTAW AMARE**

Accepted by the Faculty of Informatics, St. Mary's University, in partial fulfillment of the requirements for the degree of Master of Science in Computer Science

**Thesis Examination Committee:**

---

Internal Examiner

Full Name Shimelis Tamiru (PhD candidate) Signature----- Date-----

---

External Examiner

Full Name Dr. Minale Ashagrie Signature---------- Date-----

---

Dean, Faculty of Informatics

Full Name-----Signature----- Date-----

February, 2024

## **DECLARATION**

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

**SABA SITOTAW AMARE**

---

Signature

Addis Ababa

Ethiopia

This thesis has been submitted for examination with my approval as advisor.

**DR. MULUGETA ADIBARU**

---

Signature

Addis Ababa

Ethiopia

February, 2024

## **ACKNOWLEDGEMENTS**

I would like to thank my advisor Dr. Mulugeta Adibaru for giving helpful Commenting and supporting to my work I am also thankful to my instructors Shimelis Tamiru, coworkers, friends, for support and encouragement. Beside that I would say thank for my little Sister for supporting me during a time of my work specially data gathering and preparation time.

# Table of Contents

DECLARATION .....	i
ACKNOWLEDGEMENTS .....	ii
Table of Contents .....	iii
List of Abbreviations .....	v
List of Figure.....	vi
List of Table.....	vii
Abstract .....	viii
1 Introduction .....	1
1.1 Background .....	1
1.2 Motivations.....	4
1.3 Statement of the Problem .....	4
1.4 Objectives.....	5
1.4.1 General Objective .....	5
1.4.2 Specific Objectives: .....	5
1.5 Scope .....	6
1.6 Significance of the Study .....	6
1.7 Methodology .....	6
1.8 Tools and Techniques.....	7
1.9 Research Design.....	7
1.10 Organization of Thesis .....	7
2 Literature Review .....	8
2.1 Overview of Language Identification .....	8
2.2 Language Identification Technique.....	8
2.2.1 Word Based Technique.....	9
2.2.2 Characters N-grams Technique.....	10
2.3 Language Classifications Method .....	11
2.3.1 Naive Bayes (NB).....	12
2.3.2 Logistic Regression.....	13
2.3.3 Decision Trees .....	14
2.3.4 FRO Statistics Classifier.....	14
2.4 Related Works .....	15
3 Data Preparation and Preprocessing .....	22
3.1 Data Collection.....	22

3.2	Data Preprocess .....	24
3.2.1	Data Cleaning.....	24
3.3	Tokenization.....	25
3.3.1	Word Tokenization .....	25
3.3.2	Character Tokenization.....	26
3.4	Stemming .....	27
3.5	Feature Extraction .....	28
3.5.1	Character N-gram Extraction.....	28
3.5.2	Word N-gram Extraction .....	32
3.6	Training and Testing Data.....	33
4	Design of Language Identification and Classification Algorithm.....	35
4.1	Design of Language Identification.....	35
4.2	Classification Methods.....	37
4.3	Algorithms: Extract Character n-gram and Word n-gram .....	37
4.3.1	TRAINING: Algorithm: Language Profile Generator.....	37
4.3.2	TESTING Algorithm: language identification .....	38
5	Experimental Results.....	45
5.1	Experimental Setup .....	45
5.2	Evaluation Metrics .....	54
5.2.1	Confusion Matrix .....	54
5.2.2	Accuracy .....	63
5.2.3	Precision.....	63
5.2.4	Recall .....	64
5.2.5	F1 Result.....	66
5.3	Comparative Analysis .....	69
6	Conclusion and Recommendation .....	71
6.1	Conclusion.....	71
6.2	Recommendation.....	73
	Reference .....	74
	Appendix.....	77
	Annex 1- Python Code: Predicting with Real Data (Out of Vocabulary Word).....	77
	Annex 2-Python Code Classification Error Analysis.....	78
	Annex3-Screenshot Python CodeMost Common Chars in Unigram.....	79
	Annex 4-Screenshot Python CodeMost Common Chars in Bigram.....	80

## List of Abbreviations

BoW	Bag of Words
DT	Decision Tree
FRO	Frequency Rank Order
GB	Gradient Boost
LID	Language Identification
LIGA	Language Identification Graph
LSTM	Long Short Term Memory
MA	Macro
ML	Machine learning
NB	Naïve Bayes
NLP	Natural Language Processing
NLTK	Natural Language Toolkit
OCR	Optical Character Recognition
PCA	Principal Component Analysis
RF	Random Forest
SGML	Standard Generalized Markup Language
SVM	Support Vector Machine
TFIDF	Frequency-Inverse Document Frequency
WE	Weighted
XML	Extensible Markup Language

## List of Figure

Figure 2.1 Shows N-gram Modeling [10].....	10
Figure 2.2 Comparisons in Different Features [13].....	12
Figure 2.3 Confusion Matrixes NB [14].....	13
Figure 2.4 Results in Logistic Regression [13].....	14
Figure 3.1 Displays Total size data per Language.....	23
Figure 3.2 Distributions of word Counts across Language.....	26
Figure 3.3 Distributions of Character Counts across Language.....	27
Figure 3.4 Zipf's plots analysis for Character N-gram Frequency per language.....	28
Figure 3.5 Zipf's plots analysis for the top 5 character N-gram per language.....	29
Figure 3.6 Most Common Unigram Chars per Language.....	30
Figure 3.7 Most Common Bigram Chars per Language.....	31
Figure 3.8 Zipf's plots analysis for Word N-gram Frequency per language.....	32
Figure 4.1 General Design of Language Identification Flow.....	36
Figure 5.1 Structures of the Experimental Process.....	45
Figure 5.2 Results in NaïveBayes Models.....	47
Figure 5.3 Result in Logistic Regressions Models.....	48
Figure 5.4 Results in Random Forest Models.....	49
Figure 5.5 Accuracy rates of Decision Tree Models.....	51
Figure 5.6 Accuracy rates of Gradient Boosting Models.....	52
Figure 5.7 Accuracy rates using analyzer='Char' and N-gram range= (1, 3).....	53
Figure 5.8 Screenshot of NB classifier result on 1% Unigram & Bigram Mixture.....	54
Figure 5.9 Screenshot of Logistic Regression result on 1% Unigram & Bigram Mixture.....	55
Figure 5.10 Screenshot of Random Forest result on 1% Unigram & Bigram Mixture.....	56
Figure 5.11 Screenshot of Decision Tree result on 1% Unigram & Bigram Mixture.....	57
Figure 5.12 Screenshot of Gradient Boosting result on 1% Unigram & Bigram Mixture.....	58
Figure 5.13 Screenshot of Logistic Regression result on analyzer="char" n-gram (1, 3).....	60
Figure 5.14 Screenshot of Twenty Feature sets are used as Column.....	61
Figure 5.15 Screenshot of Random Forest Classifier Confusion Matrix result.....	62
Figure 5.16 Screen Shoot Result using analyzer='Char' and N-gram range= (1, 3).....	68
Figure 5.17 Screen Shoot Result on Twenty Feature sets.....	68
Figure 5.18 Accuracy Comparison of Different Classifier.....	70



## List of Table

Table 2.1 Shows Different Languages, used Techniques, Methods and Result.....	20
Table 3.1 Word tokens and average Word Length in our Corpora.....	26
Table 3.2 Character tokens and number of unique words in our Corpora.....	27
Table 3.3 Shows Total Corpus size used to Test and Train model.....	33
Table 3.4 Show corpus size in sentence level Test and Train model per Language.....	34
Table 3.5 Show corpus size in word level Test and Train model per Language.....	34
Table 5.1 Test Result using Naïve Bayes Models.....	47
Table 5.2 Test Result using Logistic Regressions Models.....	48
Table 5.3 Test Result using Random Forest Models.....	49
Table 5.4 Test Result using Decision Tree Models.....	50
Table 5.5 Test Result using Gradient Boosting Models.....	51
Table 5.6 Total Test result of using analyzer='Char' and N-gram range= (1, 3).....	52
Table 5.7 Test Result of using Random Forest Classifier.....	53
Table 5.8 NB classifier Confusion Matrix result on Fig 5.8.....	55
Table 5.9 Misclassified Records from Table 5.8.....	55
Table 5.10 Confusion Matrix of Logistic Regression Classifier result on Fig 5.9.....	56
Table 5.11 Misclassified Records from Table 5.10.....	56
Table 5.12 Confusion Matrix of Random Forest Classifier result on Fig 5.10.....	57
Table 5.13 Misclassified Records from Table 5.12.....	57
Table 5.14 Confusion Matrix of Decision Tree Classifier result on Fig 5.11.....	58
Table 5.15 Misclassified Records from Table 5.14.....	58
Table 5.16 Confusion Matrix of Gradient Boosting Classifier result on Fig 5.12.....	59
Table 5.17 Misclassified Records from Table 5.16.....	59
Table 5.18 Confusion Matrix of Logistic Regression Classifier result on Fig 5.13.....	60
Table 5.19 Misclassified Records from Table 5.18.....	60
Table 5.20 Random Forest Classifier Result on Fig 5.15.....	62
Table 5.21 Misclassified Records from Table 5.20.....	62
Table 5.22 Precision Result.....	64
Table 5.23 Recall Result.....	66
Table 5.24 F1 Score Result.....	67
Table 5.25 Comparison of Different algorithm with Different feature set.....	69

## Abstract

Text identification is an automatic recognition task that seeks to determine a word's meaning based on its context from the specified text in a targeted language. In richly resourced languages, this issue has been thoroughly examined and analyzed like European, but not in low resourced language especially Ethiopian language so, to mitigate such issues many researchers propose a language identifier system and now become the main research topic of many researchers. To solve the above problem propose a language identifier system, by exploring the three experiment with the first Unigrams, Bigrams and Mixture of both and second experiment analyzer='char' and n-gram range= (1, 3), last experiment twenty feature sets used as a column in the first experiment, for all classifiers, employed a unigram (n=1) feature set with four specific language instruction classes for Hadiyya, Wolaytta/Wolaytegna, Somali & Sidama on this experiment in the Naïve Bayes model, the average classification accuracy for all language was 81%, and 85%, 90%, 79%, and 89% for Logistic Regression, Random forest, Decision Tree, and Gradient Boosting classifiers and in 1% mixture of Unigram & Bigram was an average classification accuracy of the Naïve Bayes, Logistic Regression, and Random forest, Decision Tree, Gradient Boosting classifiers was 95.25%, 96.7 %, 97.56%, 91%, and 96.6%, respectively. In 60% mixture of Unigram & Bigram feature set for all classifiers with four targeted language classes, Naïve Bayes is, Logistic Regression, Random forest, Decision Tree and Gradient Boosting classifiers showed an average classification accuracy of 91% and 94% ,95.96%,88.36% and 94.87% respectively. When using n-gram range= (1, 3) analyzer='char' Logistic regression has an overall average performance of 98.9% Out of all the classifiers, this one has the highest rate and for each language Hadiyya, Sidama, and Somali wolayta is 99%, 98%, 100%, and 99% respectively. In the third experiment, twenty Sets of features were employed as a column for each model; the average rate of correct classification using Naïve Bayes is 59.71%, whereas the rates for Logistic regression, Random Forest, Decision Tree, and Gradient Boosting are 70.41%, 78.11%, and 76.69%, respectively.

**Keywords:** Language Identification, Multinomial NB and DT, RF, Gradient Boost.

# Chapter One

## 1 Introduction

### 1.1 Background

Language detection and identification is a natural language processing task where we need to identify the language of a text or document. A few years ago using machine learning for language identification was a difficult task because there was not a lot of data on languages, but with the availability of data with ease, several powerful machine learning models are already available for language identification and detection [1] [2]. As a human being, you can easily detect and identify the languages you know for example, we can easily identify Latin transcriptions like Hadiyya, Wolaytta/Wolayteгна, Somali & Sidama and other language Amharic but it is also not possible to identify all languages for all humans unless a mother tongue or learned a language. This is where the language identification task can be used google translate is one of the most popular and power full language translators in the world which is used by so many people around the world. It also includes a machine learning model to detect languages that you can use if you don't know which language you want to translate [3].

The earliest known work to describe a functional LI program for text examined multiple discriminated analyses to teach a computer how to distinguish a language at the word level, between "English, Swedish and Finnish". The researcher compiled a list of linguistically-motivated character-based features, and trained his language identifier on 300 words for each of the three targeted languages [3]. The training procedure created two discriminates functions, which were tested with 100 words for each language for English, Swedish and Finnish. The experiment resulted in 76% of the words being correctly classified; even by current standards this percentage would be seen as acceptable given the small amount of training material, although the composition of training and test data is not clear, making the experiment un reproducible [3].

The aims of language identification are to mimic human ability to recognize and distinguish specific languages. Over the years, a number of computational approaches have been developed that the computational techniques are based on statistical methods and require curate examples, through the use of specially-designed algorithms and indexing structures, are able to infer the language being used without the need for human intervention [4]. The capability of such systems could be described as super-human: an average person may be able to identify a handful of languages and a trained linguist or translator may be familiar with many dozens, but most of us will have, at some point, encountered written texts in languages they cannot place. However, LI research aims to develop systems that are able to identify any human language a set which numbers in the thousands [2].

Till now Research on different language identification have been employed in a variety of approaches the major approaches include: detection based on stop words usage, detection based on character n-grams frequency, detection based on machine learning (ML) and hybrid methods. Many standard machine learning techniques has been applied to automated text categorization problems, such as Naïve Bayes classifiers, support vector machines, n-gram frequency rank order, and neural networks classifiers are mentioned [5].

Text language identification and detection plays a major role in several Natural Language Processing applications[6]. It is mostly used as an important preprocessing step. The application of language identifications includes e-mail routing and filtering engines, text mining applications, identification of the language or encoding of WWW pages, information retrieval systems, content based and language specific web crawlers and search engines and spell checker applications [7].

The main challenges in language identification and detection include: Improving the coverage of language identification systems by increasing the number of languages that systems are able to recognize, Improving the robustness of language identification systems by training systems on multiple domains and various text types, Handling non-standard texts (e.g. multilingual texts, computer- mediated communication content, code-switching), and discriminating between very similar languages, varieties and dialects [5].

This study focuses on addressing the challenge of identifying languages within a text that may consist of content from multiple languages in the dataset. In this section, we present a technique for detecting and determining the language of a given text, as well as depicting the accuracy of each model employed and prediction result or outcome for targeted language and also summarized by measurement matrix counting accurate and inaccurate prediction and shows actual and prediction language.

The study demonstrates the problem of language identification for the Cushitic and Omotic on low and under-resourced Ethiopian languages. We provide a LID model capable of accurately identifying four Ethiopian languages with manually annotated dataset and trained a number of systems on this dataset and apply Multinomial NB, LR DT, RF, and GB classifier for LID of any length of a text and also compare existing established LI methods with character analyzer or word n-gram (1, 3) features, unigram and bigram mixture of text as well as twenty features set as a column.

The algorithm was evaluated using recent measurement metrics and our best results have been obtained from a system using character analyzer or word n-gram (1, 3) features in Logistic Regressions and in Random forest classifier 1% mixture of Unigram & Bigram used as a feature set achieved the highest score, beside that the study provide detailed error analysis of the system, and also be possible to predict with the real data or out of vocabulary words in addition to this it can be possible to predict in word level, sentence and paragraph level of a document. Finally from this study I observe the challenges with high close related similar language thus, Identifying related Ethiopia language makes the identification task more difficult because the languages themselves are relatively similar.

## **1.2 Motivations**

Ethiopian language identification from text data primarily revolve around the need for effective language processing and understanding in a multilingual society. This linguistic diversity poses challenges in various areas, such as education, communication, language planning, and policy-making. Language identification from text data can help in accurately determining the language of a given text, which in turn can contribute to addressing these challenges and promoting effective communication among different language communities.

Additionally, language identification is important for preserving and promoting Ethiopian cultural heritage, as it allows for the development of tools and resources for analyzing and understanding the various languages used in the country and also no previous research has been conducted on language identification specially Hadiyya, Wolaytta/Wolayteгна so, motivated by this gap in the literature and its practical impact developing language identification model for under-resourced languages Hadiyya, Wolaytta/Wolayteгна, Somali & Sidama. These languages used as official working language and have many speakers in total, now a day with growing access to technology and the internet, in all language an increasing portion of the native speakers is consuming and producing digital content and motivating the NLP applications to cope with the demand.

## **1.3 Statement of the Problem**

Due to the increasing amount of data being generated and stored online, digital documents continue to grow in size and complexity, the need for accurate language identification becomes increasingly important. Language identification is crucial for a variety of applications, including search engines, machine translation, text analytics, and content categorization and the growing volume of multilingual content on the internet, social media, and other digital platforms underscores the importance of robust language identification systems. These systems must be able to accurately identify and process a wide range of languages, dialects, and writing systems to effectively manage and analyze the vast amounts of digital information available.

Beside that Language identification is crucial for the localization of digital content, including websites, applications, and user interfaces by accurately identifying the language of the user, personalized content and experiences can be provided, enhancing user engagement and satisfaction. However, no previous research has been conducted on language identification

model specially Hadiyya, Wolaytta/ Wolayteгна So, developing a language identifier model for selected Ethiopian languages Hadiyya, Sidama, and Somali wolayta using Hybrid approach is one aim of this research. As well as the previous research has a lack of comparative analysis means it shows a comparative gap and methodological gap and the previous study failed to provide predicting with the real data or out of vocabulary words and also lacks detailed error analysis so, this study tries to solve this research gap by implementing different techniques and approach for our domain of language.

The Research is intended to get Answers for the Following Research Questions.

**RQ1.** Which dataset is presently accessible for my intended use?

**RQ2.** How the preparation and preprocessing of the dataset will be addressed?

**RQ3.** How the extraction and selection of relevant features will do in each language?

**RQ4.** How the language identification model will be designed?

**RQ5.** Which evaluation measurement techniques were used for a model performance?

## **1.4 Objectives**

### **1.4.1 General Objective**

The objective of the research is building a language identification models for Ethiopian Cushitic and Omotic Languages using Hybrid approach.

### **1.4.2 Specific Objectives:**

- To review the Current state of the art literatures on selected Ethiopian language.
- To Collect and preprocess our data set and experiment language statics data.
- To extract features and select the unigram and bigram of relevant features in each language.
- To Design and implement a machine learning model in order to carry out a comparative analysis.
- To evaluate identification performance of each model.

## 1.5 Scope

The scope of this study only the four languages found in Ethiopia to differentiate and predict the language of a given sentence from the four languages which are Hadiyya, Wolaytta/Wolayteгна, Somali & Sidama To achieve the objectives of the study, different machine learning approaches and statistical approaches are used (Multinomial NB, LG ,DT, RF, GB)Because the lack of document under resourced language and the time constraint my thesis is limited to the above language.

## 1.6 Significance of the Study

- Document is initially for the use of reference in research for the particular domain.
- Provide the existing knowledge and understanding in a particular field or topic.
- The paper provides valuable insights and analysis that can help further develop theories, guide future research work.
- Evaluation of the algorithm against existing approaches and test sets provides valuable benchmarking in comparison for language identification research.
- The findings of this research can have practical implications for various applications, such as language processing, machine translation in the context of Ethiopia language.

## 1.7 Methodology

Methodology followed by on this thesis is:-

- Data set: a text of Hadiyya, Wolaytta, Sidama, and Somali collected from different linguistic sources such, Religious books (Bible, Quran), academic books, Proclamation.
- Data analysis: use different technique to clean our corpus unwanted records symbols, Null Values, Stop words, lowercase, punctuation mark normalization and steaming.
- Technique: Characters  $N$ -grams technique, word based and character based approach used ,Unigrams, Bigrams and Mixture of both and use two techniques of features extraction TF-IDF advanced count vectorizer technique to convert text data into a form of vector as an input and BoW techniques to compile a list of commonly known terms from our corpus.
- Experiments: a monolingual language identification and accuracy with different techniques and different approaches are tested and compared.



- Evaluation: follow a standard and advanced evaluation measurement like “precision, recall and F1” measure to compare the baseline of proposed classification techniques.

## **1.8 Tools and Techniques**

Language identification from text data involves a variety of tools and techniques that help to determine the language of a given piece of text. Most common tool used is language detection libraries; these are sets of pre-built functions that can automatically detect the language of a given text. Some popular language detection libraries include: Pandas Library (for Data Manipulation), NLTK (Natural Language Toolkit) Numpy, seaborn, and matplotlib.

## **1.9 Research Design**

The research design used in the study is an experimental research design, which involves an experimental approach to address the objectives of the paper. This typically includes the use of machine learning algorithms, natural language processing techniques, and statistical analysis to identify patterns and features within the text data that can be used to determine the language by leveraging computational and statistical approaches used.

It includes decisions about the type of data to be collected, the methods of data collection, the tools and techniques for analysis, and the overall approach to be taken in the research study. This could involve considerations such as selecting appropriate text data sources, choosing the right machine learning algorithms for language identification, and determining the criteria for evaluating the accuracy of language identification. So the research design is crucial for ensuring the validity and reliability of the findings in this field.

## **1.10 Organization of Thesis**

The first chapter is an introduction part to the problem area of text language identification for Ethiopian Language, general objectives and specific objectives the remaining chapter are chapter two is literature review shows a general overview of Text language identification approaches, techniques, challenge and related work presents the third chapter dealt on Methodology, Tools, Data Preparation and Processing four focuses on Design and classification algorithm and the last Chapter five Experimental Results/Evaluation Metrics and finally Conclusion and Recommendation is present.

## Chapter Two

### 2 Literature Review

#### 2.1 Overview of Language Identification

Language identification system gives at each document's extracted text to identify the major language and up to two secondary languages. This allows you to see how many languages are present in your collection, and the percentages of each language by document. You can easily separate documents by language and batch out files to native speakers [\[6\]](#).

Language Identification as a task predates computational methods the earliest interest in the area was motivated by the needs of translators, and simple manual techniques have been created to swiftly locate documents in particular languages [\[2\]](#).

The first publication that describes a text-based functional language identification program used multiple discriminate analyses to teach a computer how to distinguish, at the word level, between language like “English, Swedish and Finnish”. The researcher compiled a list of linguistically-motivated character-based features, and trained language identifier on 300 words for each of the three target languages and the training procedure created two discriminate functions, where by each language's test consisted of 100 words [\[2\]](#).

As the result 76% of the words were correctly classified as a positive in the experiment; even by today's standards, this percentage would be considered acceptable given the limited training material, although the composition of training and test data is not clear, making the experiment un reproducible[\[2\]](#).

#### 2.2 Language Identification Technique

Language Identification Technique is the most important part in natural language data per processing task [\[8\]](#). To determine the language of a document, or at the very least to infer the family language, numerous parameters can be employed. The document's identification can be carried out at many levels, with each level's reliability either being good or bad.

Language identification algorithms assumed every document is to have been written in one of the known languages for which training data is present and is thus defined as the process of selecting the most excellent matching dialects from the set of preparing languages[2].

## **2.2.1 Word Based Technique**

### **2.2.1.1 Short Words**

It is one of the statistical language modeling word-based approaches are called short word-based approach. It solely constructs the language model using words up to a predetermined length and is independent from the actual word frequency. Common limits are 4 to 5 letters. By exploring one million characters of text for each language, tokenizing them and extracts all words with a maximum character length of five that occurred at least three times. The idea behind this technique is the language specific significance of common words like conjunctions having mostly only marginal lengths. Depending on the language, on his language models contain between 980 and 2750 Authors, it observes that short words will outperform as good as group of function words [4].

### **2.2.1.2 Word Frequency**

This is another type of the statistical language modeling it is the simplest method for creating a language models and use words from all languages in the training dataset. As stated by the Zipf's Law, it uses the words that appear the highest and most frequently. The frequent words method obeys Zipf's Law, where language models are generated based on specific amount of words, having all words appear in a text or document with the highest frequency possible. Numerous pieces of work have been done in the most frequently 100 words, and generate a language model by using the 1,000 words with the highest frequency of words and considers 100 of the most common word in each language extracted from training data for nine languages and 91% of all Documents were accurately classified. Beside that there is no indication in the paper about the frequency used; either relative or absolute examined one hundred most frequent words in word frequency tables where every word gets a normalized frequency value[4].

### 2.2.1.3 Dictionary of Unique Words

Unique word dictionaries include only those words of the language that are not part of other languages targeted by the language identifier[2]. For the purpose of linguistic distinction the researcher employs unique short words between one and three characters a glossary of unique word was used. The researchers evaluate Language variety specific word lists were part of those features with feature union in scikit learn; they carried the same weight as character flood counts.

### 2.2.2 Characters N-grams Technique

It is a statistical language modeling and successful approach for generating language models. for text categorization and found out that it also did well when it came to identifying languages , in this technique, from a corpus of documents, a language model is generated using N-grams instead of complete words that are used in the first two approaches. An n-gram is a contiguous N-character slice of a string or a substring of a word and respectively words depending on the size of N. The underscore character is frequently marked to indicate the beginning and end of words or spaces before N-grams are created [9].

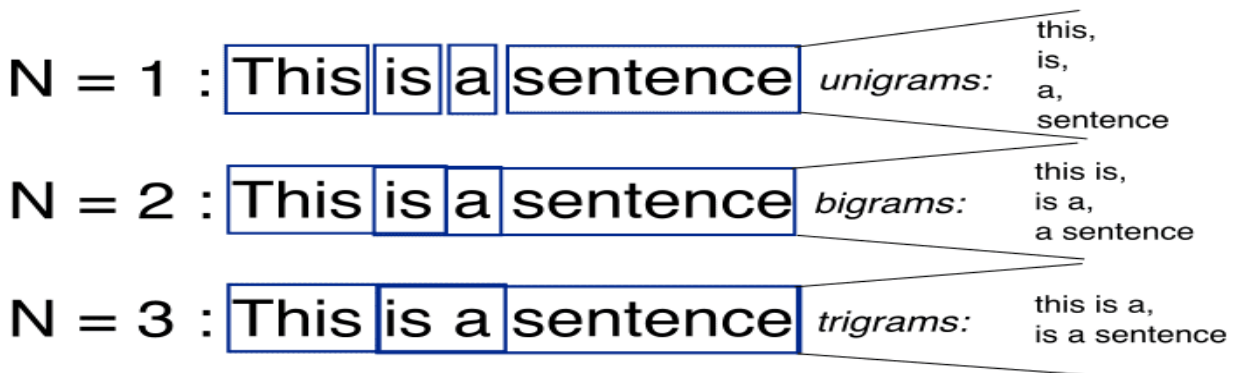


Figure 2.1 Shows N-gram Modeling [10]

#### 2.2.2.1 Character N-Gram

Language identification, statistical models can be created on the basis of the text's word count or letter count in the given text), N-gram statistics or a combination of the two[11]. In the literature, the most common statistical method is the character-based N-gram model, which is for short text fragments, is preferable to the word-based model and performs equally well on large fragments.

It doesn't require prior understanding of a language making it highly accurate and easily applicable to any given text. Hence, most LI systems use character N-grams an N-gram is a sequence of n consecutive letters. The N-gram-based approach for LI divides the text into character strings of equal size. Some languages are assumed to use certain N-grams more frequently than others[11].

#### **2.2.2.2 Graph-Based Technique**

Language Identification in Graph-based N-gram Approach (LIGA) described with N-gram occurrences and presences, they also take into account their ordering, and they use tagged data to develop a graph language model. The frequency of the trigrams are represented by the weights of the nodes and the weights of the edges capture transitions from one character trigram to the next to develop a language model use a training corpus of that language's literature they compute the transitions and frequencies of trigrams and divide these counts by the total number of nodes or edges in the language[9].

### **2.3 Language Classifications Method**

This is the second stage of the language identification process the document's language is determined by employing model the produced document as input for the classification technique, language identification can be done using a variety of text categorization techniques [4].

To classify a source document using selected language models to determine and calculate the distances among them. The targeted language that is closest to the source document in terms of distance is selected as the document's language. However, for each classifier, a separate method is used to calculate distances; the initial steps mostly remain the same after preprocessing, several N-grams and the frequency of their occurrence are taken from the training corpus for every language, by using the natural logarithm, these counts are converted of them, add one and divide each value by the dataset's highest count [9].

The researcher mentions it is one of the simplest approaches to text classification and by observing the frequencies of n-grams, a model can be developed associated with different classes present in the text, and the language of the sentence can be predicted [12].

### 2.3.1 Naive Bayes (NB)

Examined a training Multinomial Naive Bayes model, produced a baseline result because it prototypes and executes quickly and known to provide decent results in the field of text processing. No pre-processing of the text commonly done in the field like stemming or stop word removal because we believe that could potentially remove important signatures of a particular language, particularly when two geographically separate groups of individuals speak the same language (e.g. Portuguese spoken in Portugal and Brazil). character and word n-grams were used in his experiments[13]. The character n-grams turned out to be particularly useful when compared to word level n-grams, character level n-grams behave very differently as shown in Fig.2.2 Single characters carry little information and therefore character N Gram's performance improves quite sharply as the number of characters is increased before saturating at about n=8. The researcher experiments with character n-gram in both restricted at word boundaries and spanning across word boundaries. The latter has a marginal performance boost at the cost of longer training time and memory pressure. On his experiment the performance of the character-level and word-level n-gram models is comparable[13].

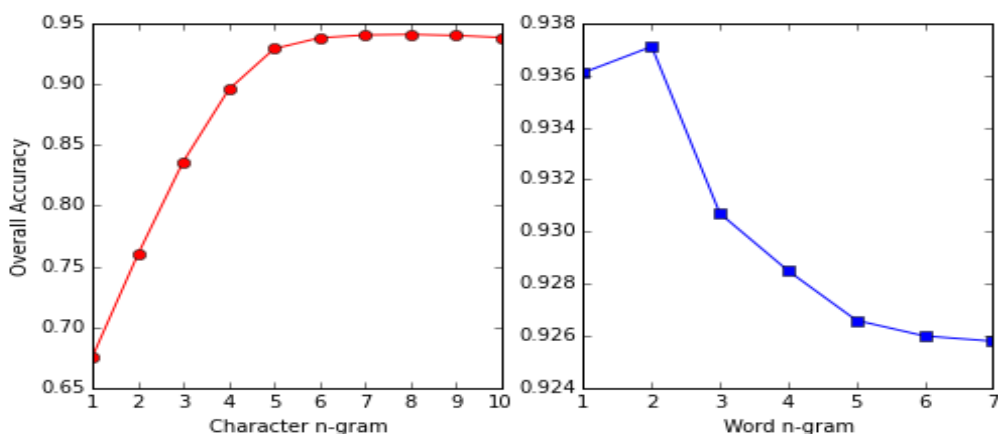


Figure 2.2 Comparisons in Different Features [13]

The Multinomial Naive Bayes algorithm is appropriate for classification using distinct features using the Bayes theorem, the classifier makes an estimate as to what a text's tag is determines the likelihood of each tag for a given sample and output the tag with the highest probability [14]. The Multinomial Naive Bayes is widely employed to allocate documents to classes in

accordance with a statistical examination of their contents in this instance, a dataset with 17 languages is used to train the Multinomial Naive Bayes classifier model including” English, French, Spanish, Portuguese, Italian, Russian, Swedish, Malayalam, Dutch, Arabic, Turkish, German, Tamil, Danish, Kannada, Greek, and Hindi”. By this it evaluates the Multinomial Naive Bayes classifier model that has been trained and the accuracy is calculated to be 97.87% [14].

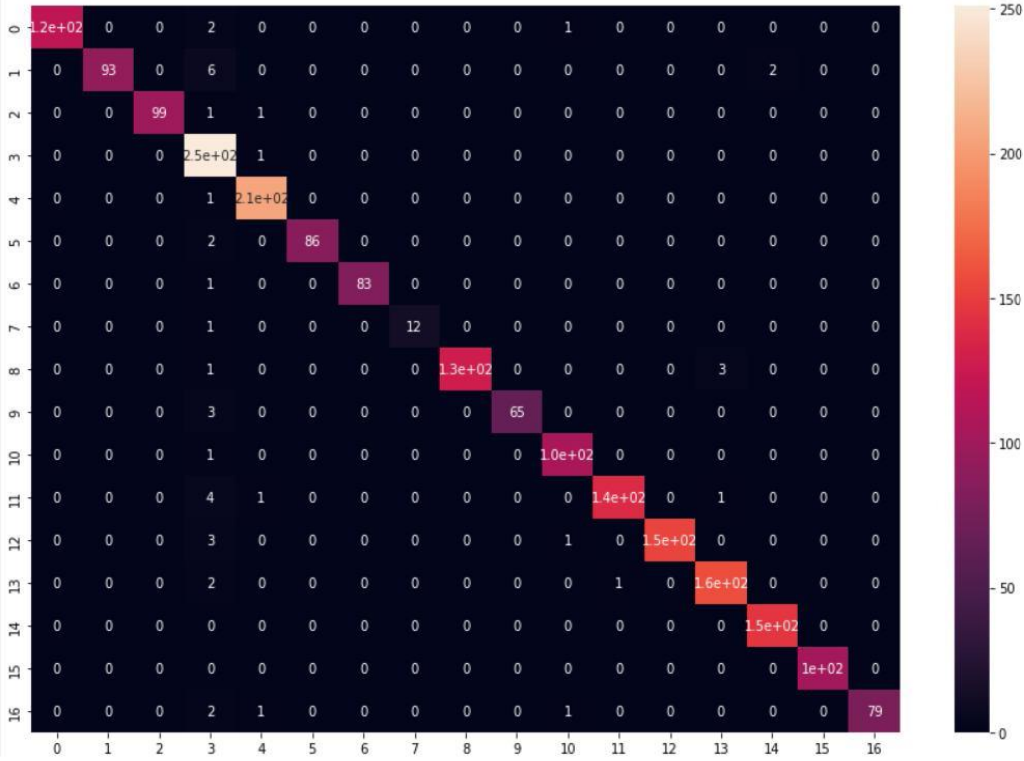


Figure 2.3 Confusion Matrixes NB [14]

### 2.3.2 Logistic Regression

In this logistic regression algorithm shows the word n-grams performed worse than the character level n-grams. The model was able to perfectly fit the training set, as seen in Fig. 1.4 but the performance on the validation set plateau close to 0.945 on this A character 9-gram model had the best performance. That has all n-grams up to n=9 on this no consecutive words were captured by these n-grams [13].

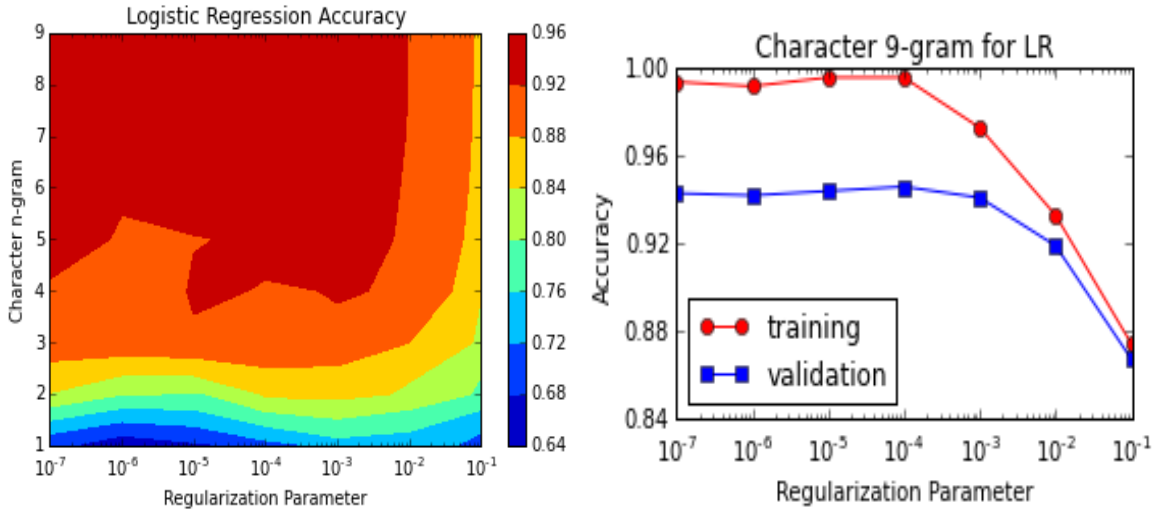


Figure 2.4 Results in Logistic Regression [13]

### 2.3.3 Decision Trees

The researcher used DTs without frequency information based on characters and their context each node is divided into child nodes when the DT is being trained [2]. For each node feature is chosen to optimize the information gain at that node, a. The information gain is calculated for each feature and node chooses the feature with the greatest gain. In the identification phase, the nodes are traversed until only one language is left (i.e. a leaf node is reached). He used DTs with character trigrams and was successful in obtaining an F-score of 68.76 at the word level recognizing the distinctions between the 11 South African languages.

### 2.3.4 FRO Statistics Classifier

To identify a document's language, employ a method that determines an out-of-place measurement for each N-gram in the document models it calculates the separation between two N-grams of the document model and the different language models this method is often referred to as rank-order statistics [9].



## 2.4 Related Works

There are several scientific research works and studies across the world related to text language identification for resourced language and low resourced language but it is lack on investigation under resourced language like African language for implementing A model or artificial intelligent . The following literatures are closely related to my study.

Text categorization involves automatically classifying and provided textual content (such as paragraphs or documents) into predefined categories. Language classification has gained significant research attention due to the exponential growth of digital texts, which necessitates the automatic organization and indexing of extensive text repositories in diverse manners. These methods are presently used in various fields, such as identifying spam, determining language usage, attributing authorship, classifying text genres, identifying topics, and categorizing subjective sentiments [\[7\]](#).

As indicated in [3]the researchers examined a hybrid technique that combines character and word n-grams with a rule-based approach for Ethiopian languages. The language identification method aims to ascertain the language utilized in a text across different levels such as single words, medium word, and large documents. This is for both single-language and multi-language scenarios.

The average F-measure for Amharic, Geez, Guragigna, and Tigrigna, respectively, is “70.39%, 76.95% 4, 73.69%, and 78.98%,” according to The hybrid approach integrates fixed-size character n-grams with Location information, word n-grams, and a method based on rules On average, it attained F-measure scores of around (“ 83.57%, 84.53%, 86.67%, and 87.44% for Amharic, Geez, Guragigna, and Tigrigna”) respectively is shown by the hybrid of infinite n-grams with location, word n-grams, and rule-based approach. The accuracy of the hybrid model has increased to “99.85%, 99.74%, 100%, and 99.93% for Amharic, Geez, Guragigna, and Tigrigna, “respectively. In multi-language scenarios, it also accomplishes an average F1 score of 100% for both medium word, and large documents evaluations, but for single word it reaches an average F-measure of (“82.64%, 86.38%, 87.19% and 86.81% For Amharic, Geeze, Guragigna and Tigrigna “) respectively Thus, it is evident that integrating medium word, and large documents restructuring into the hybrid model of infinite n-grams with location feature set represents an optimal result. The current study may not have sophisticated feature selection

techniques to discriminate between very closely related languages and not provided comparative studies with different machine learning classifiers to determine the best one for the proposed language identifier.

The work in [15] wrote his master's thesis in seven primary languages of Ethiopia were utilized in his investigation, including, "Afar, Amharic, Nuer, Oromo, Sidamo, Somali and Tigrigna" his aim was employ the Dictionary Method, SVM classifier, and Naive Bayes classifier with a technique of generating character n-grams with a size of (n=3 ) to train naive Bayes and SVM classifiers. Using a classifier with size (n=3), such as NB or SVM a mean classification accuracy of was "98.37%, 99.53%, respectively and the dictionary method demonstrated a mean classification accuracy of 90.53% ".with a size of (n=3) NB and SVM classifiers demonstrated a mean classification accuracy of "95.16% and 96.2%" respectively. To evaluate multi-language scenarios, this corresponds to 95.71% accuracy. The current study may not have fully explored the potential of combining classification methods with linguistic features.

This researcher [7] addressed issues related to corpus sources for the purpose of text LID in Afaan Oromo, Afar, Sidama, and Somali languages, languages written with the Latin alphabet to the size of corpus required to identify text documents of the languages, for his dissertation a comparison was made between n-gram FRO and NB for identifying a targeted language for Ethiopian Cushitic languages.

Beside that the corpus for the study was collected from sources such as TV news websites, Bible, news bulletins, government documents, and documents from ministry of education to insure the Corpus spans various domains .Web Corp tool was used to collect corpus from news web sites For the selected model evaluate Documents of sizes 15, 100, and 300 characters windows were used. For test string of size 15 characters accuracy of 99.55% on character n-gram feature set and 99.78% on character n-gram feature set was achieved for Naïve Bayes classifier. The identification accuracy rate of NB for both FS when the test 100 characters is 100%. For test string of size 300 characters frequency rank order as a classifier, accuracy of 63.55% ,accuracy of 86.78% was attained in identifying char n-gram with their corresponding location information within a word FS. The gaps of the paper was it does not mention any evaluation metrics or performance measures used to assess the accuracy and effectiveness of the language identification models and the paper lacks information on the computational resources or

hardware used for training and testing the models, which could impact the scalability and efficiency of the proposed approach.

The paper presents by [16] in his research, a hierarchical NB and lexicon-based classifier were utilized to perform his aim of (LID) on Phrase level used for Low resourced languages. The algorithm's performance was assessed on Phrase level from the eleven South African languages, some of which are closely related. It was observed that the accuracy of the proposed algorithm relies significantly on the availability of support from the lexicon. Additionally, it was noted that without a robust lexicon, a non-stacked naive Bayesian classifier might even outperform the proposed approach. The study was lack comprehensive comparison to all recent approaches and it does not provide a detailed discussion of how to address these challenges for language identification in South African languages.

As per the study conducted by [17] for Indian languages like “Tamil, Hindi, Kannada, Malayalam “three machine learning algorithms were evaluated in conjunction with two vectorization methods. The machine learning algorithms included NB, LG, and SVM Classifier. Vectorization methods were utilized for extracting features from the data it is the (TFIDF) and (BoW). For this the highest precision attained was 98.45% (via the LR with TF-IDF technique), trailed by 97.34% (BoW technique and NB). Each of this classifier exhibited distinct performance when subjected to various feature extraction methods used. It was noted that NB exhibited superior performance compared to other techniques when using the BoW feature extraction method. These research gaps were not to handle short sentences and multilingual documents effectively so the need for further investigation.

In this study [1] the context discusses various methods for automated language identification of written text, including those based on character n-grams and dictionaries. It proposes a new dictionary-based method that constructs language models using word relevance instead of common words. The method aims to address issues with identifying very short texts, handling unknown languages, and identifying multiple languages within a single text. It describes evaluating the method on various languages and segmented texts. The paper was Limited in analysis of results.

A study by [18] employed a machine learning algorithm for the purpose of recognizing the language used in online content, including short messages, comments, and posts. It is used

a(LSTM) A machine learning model was constructed and compared to Face book's Fast Text. The findings indicate that the LSTM algorithm attained an approximate result of 95%, whereas the fast Text achieved performance of 97%. However, the LSTM algorithm had difficulty text identification in that was below fifty characters long than it did with longer texts. Additionally, LSTM algorithm's performance in classifying data was relatively poor when it came to languages that were similar, such as Croatian and Serbian. Despite this, both the LSTM and fast Text models achieved accuracy rates exceeding 94%, which is considered high. The thesis highlights a research gap in the availability of high-quality, up-to-date, and accurately annotated datasets for training and testing LID models.

This paper presents [19] research on the challenge of identifying speakers native language he employed a machine learning algorithm created a machine learning system for the purpose of determining the primary language of native speakers who have written English texts despite not being native English speakers, His used approach involves the use of an SVM learner, resulting in achieving an 82.4% accuracy with just 55 features. The paper focuses on the use of language modeling and cross-entropy scores as features for supervised learning. Paper limitation was it does not explore other potential features or techniques that could potentially improve the accuracy of native language identification.

The purpose of the investigation conducted in the paper [20] was to investigate the effectiveness of a linear SVM that was trained on character features. These character features were language-neutral, and the investigation involved the NLI Native language identification Shared Task 2017. The primary framework that achieved the highest effectiveness was solely employed n-grams with 1 to 9 characters as features. The evaluation dataset showed an impressive 87.56 F1-score. The gaps the paper does not discuss any potential limitations in the evaluation of features and systems used in the final submissions, which include function words, letter n-grams, part-of-speech bigrams, and error types.

As per the study conducted by [14] the identification of language is performed through the utilization of a Multinomial Naive Bayes classifier when dealing with text input. Trained on a dataset encompassing 17 different languages, it boasts an impressive accuracy rate of 97.87%.The paper was does not provide information on the dataset used to train the Multinomial Naive Bayes classifier model.

The study [21] discusses an approach to distinguish between several Indian languages that contain a combination of different scripts in a social media such as “Bengali” ,”Gujarati”, ”Hindi, “Kannada”, ”Malayalam”, “Marathi”, “Tamil” and “Telugu”. A two stage classification approach is used, where the first stage identifies the sentence level and the second stage performs word level classification to identify the language of each word. Various machine learning classifiers such as Naive Bayes, MaxEnt, and SVM are evaluated. Adding more training data is found to improve the accuracy of sentence level classification. An overall weighted F1 score of 0.7692 is reported for the test run submitted to the shared task. The gaps of this paper are reporting results, lacks detailed error analysis, comparison to alternatives, and discussion on generalizability and limitations associated with the n-gram-based method.

The researcher [22] discusses a technique for identifying the script or language of text in multilingual documents. It presents Probabilistic Neural Network (PNN) and K-Nearest Neighbors (KNN) based methods for determining if text portions are in” Kannada, English, or Hindi”. Features like top/bottom profiles, max rows, horizontal/vertical lines, and shapes are extracted from words and compared to a stored database of script examples. The methods were tested on sample images, as the result PNN method achieved higher average accuracy (>95%) compared to KNN (>90%).The models have gaps the trained and tested only on word-level script samples.

This study explores [8] the utilization of character n-grams to detect complex words in “English, German, and Spanish “writings. The author employs multinomial Naive Bayes methods and utilizing character n-gram frequencies as features for the classification of words as either complex or simple. This method received moderate rankings for the shared task for complex word identification, for the “English, German, and Spanish “writings but much lower for the cross-lingual French task. The paper acknowledges limitations of short text, and does not provide detailed analysis of performance for different text lengths.

Table2.1 Shows Different Languages used Techniques, Methods and Result.

Ref	year	Features set (Technique)	Method	Language	Accuracy
[1]	2014	"character n-grams "	"dictionaries methods"	"European languages"	Promising results
[7]	2014	characters N-gram windows 15,100,300	FRO and NB	"Afaan Oromo, Afar, Sidama, and Somali"	99.55%,99.78% respectively
[19]	2015	Cross entropy scores	SVM	English	82.4%
[21]	2015	character n-grams	Naive Bayes classier, MaxEnt, Naive Bayes EM	"Indian languages namely Bengali, gujarati, Hindi, kannada, Malayalam, Marathi, Tamil, telugu."	0.7204, 0.6887, 0.7684 respectively
[22]	2015	smoothing and noise removal	KNN, PNN	Kannada, English and Hindi	90%,95% respectively
[20]	2017	1-9 character n-grams	linear SVM	Chinese japans, Korea, Hindi, French, Italy, Spain, German, Turk, tel, ARA	87.56%
[16]	2018	characters N-gram	Naive Bayesian and lexicon based classifier	11 official South African languages	-
[3]	2018	"the hybrid of infinity n-gram with location feature set"	"Hybrid of fixed-size character n-grams with location, word n-grams, and rule-based approach."	Amharic, Geeze, Guragigna and Tigrigna	99.85%, 99.74%, 100%, and 99.93%
[8]	2018	"character n-gram frequencies "	"multinomial Naive Bayes classifier"	"English, German and Spanish"	
[18]	2020	fast Text	short-term memory (LSTM	Croatian and Serbian	94%,

<a href="#">[15]</a>	2020	Character n-grams of size 3	Dictionary Method, SVM classifier, and Naive Bayes classifier.	“Afar, Amharic, Nuer, Oromo, Sidamo, Somali and Tigrigna”	98.37%, 99.53%, and 90.53% respectively
<a href="#">[14]</a>	2022	(Bag of Words (BoW)).	Multinomial Naive Bayes, Lang Detect	“English, French, Spanish, Portuguese, Italian, Russian, Swedish, Malayalam, Dutch, Arabic, Turkish, German, Tamil, Danish, Kannada, Greek, and Hindi”	97.87%.
<a href="#">[17]</a>	2022	“Frequency-Inverse Document Frequency (TFIDF) and Count Vectorizer Bag of Words (BoW)”	“Naïve Bayes, Logistic Regression, and SVM”	Tamil, Hindi, Kannada, Malayalam	98.45%, 97.34%

## Review of Summary

The results of the literature review showed most of the researcher used feature set like character n-gram features to identify language of the texts some used n-gram frequency (Ranking method), n-gram counts and word length as a feature, order of words and n-gram frequency (graph based) is used and algorithm is assessed in comparison to recent methods employing support vector machine, Naïve Bayes, the LSTM algorithm's model, rule-based approach, Decision Tree, Random Forest utilized for Language Identification of text with different lengths,. in Naïve Bayes, Logistic Regression, and SVM obtained the top result .So Based on gaps identified in the literature there is a motivation to perform investigation on the above mentioned language with different classifier and features set that helps determining the language of a provided text.

## Chapter Three

### 3 Data Preparation and Preprocessing

In this section, discuss data collection and preprocessing or analysis in 3.1 and 3.2 of the document and in part 3.3.and 3.4 discuss word and character tokenization and text steaming, the feature set or technique used on our model discussed in part 3.5 and datasets amount of corpus utilized for a model training and testing per language discussed in 3.6 part of the document.

#### 3.1 Data Collection

The data collection for this work is based on our goal it is four dataset in Somali, Hadiyya, Wolayta, and Sidama language. To prepare our data set used a variety of websites, books, an online Bible, Quran and social media platforms like Face book and Proclamation were used.

The sizes of data volume are 2MB in Total language. Because of scraping the raw data from Wikipedia and different web site which has a lot of unnecessary symbols and numbers that would lower or degrade the model's prediction quality, most of the time data that is scraped from websites not cleansed so we have to use automated preprocessing to cleanse our prepared corpus using Python script, extraction, compilation, data cleaning were one before the data test move to execution.

##### 3.1.1 Sources of Data

Data sources play a vitally important role in research and decision-making processes. The accuracy reliability of data can significantly impact the outcomes and conclusions drawn from analyses. Understanding the different sources of data is crucial for ensuring the quality of information used. This is the crucial steps for researchers to conduct meaningful and rigorous studies. For this study utilizing a secondary data sources, researchers can gather reliable information to support their research objectives and contribute to knowledge advancement. By using the Wikipedia dumps as corpora of the available languages and in addition, collect data by crawling from different websites beside that in bible and government proclamation; once the data is collected then go to preprocessing raw text.



### 3.1.1.1 Bible

The Bible which can be in soft copy or hard form written the holy scripture of the religion it is remarkable collection of ancient writings Christians believes to be god’s revelation to people.

### 3.1.1.2 Proclamation

For the aim of creating the corpus of our targeted language which can be use in soft copy form a government official working proclamation.

### 3.1.1.3 Books

It is any kind of book whether soft copy or hard copy can be utilized for this project or this one, soft copies of teaching books cultural, historical were used.

## 3.1.2 Corpus size

The corpus size in our target language refers to the amount of text data used for training and testing language identification models. Here below the corpus of our target language total size listed in the Figure 3.1.

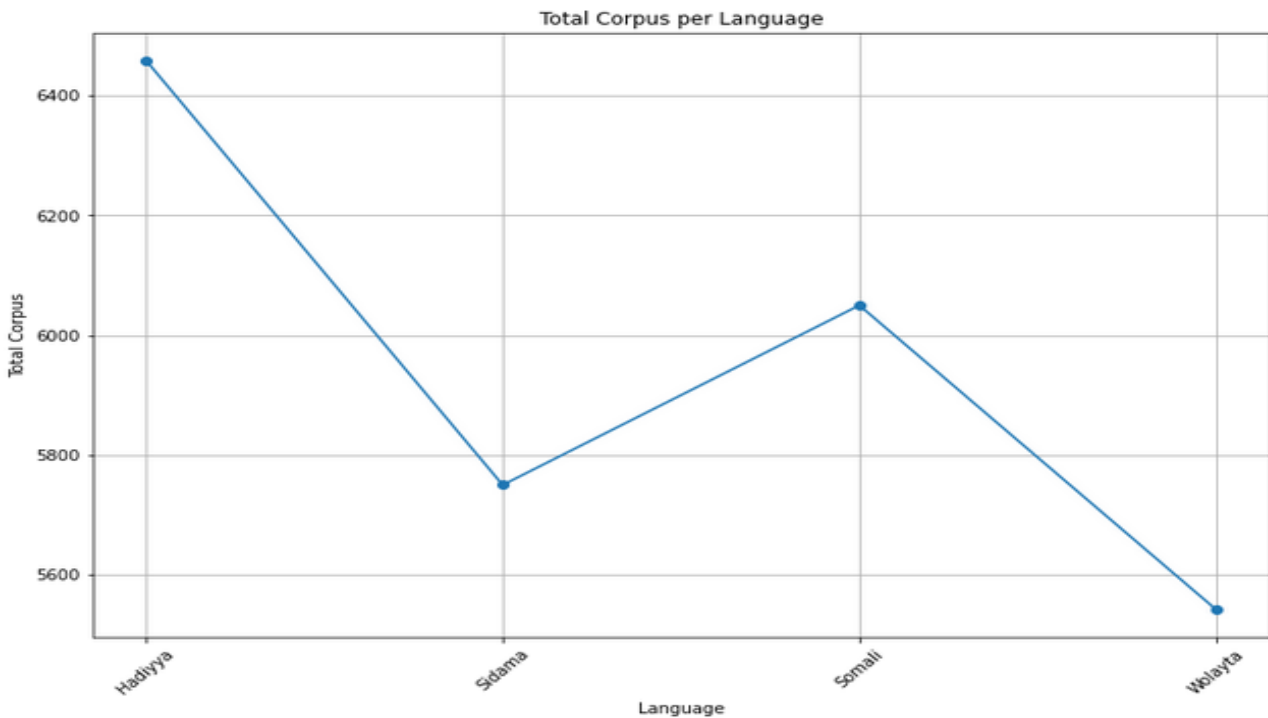


Figure 3.1 Displays Total size data per Language

## 3.2 Data Preprocess

Data preprocessing plays a crucial role in datasets for machine learning models. It involves cleaning, transforming, and organizing data to improve its quality and efficiency. A well-preprocessed dataset is the foundation for building robust machine learning models. It is obvious the real world data needs processing before feeding it to any machine learning or deep learning algorithms because when we are unable to complete this preprocessing technique performance of the employed algorithms may have declining[12].

Our dataset was prepared using scraping the raw data from Wikipedia and different web site so it is necessary to clean our data unwanted records and symbols from the language of our interest, Hadiyya, Wolaytta/Wolaytegna, Somali & Sidama corpus So use different technique and tools for this thesis the major one is Null Values, Stop words, Normalization, and removing special character and number for example hash tags and others punctuation mark and steaming.

### 3.2.1 Data Cleaning

#### 3.2.1.1 Null Values

To determine if a set of our prepared data set contains a null result or not it means checking for missing value use a panda's frame function to clean up my corpus because my corpus has little null value and prepared clean new data collection for our work otherwise, it will result in a long and ugly error notification when the model is predicting. Here below shows the result of null value from our corpora

```
In [5]: df.isnull().sum()
```

```
Out[5]: Text      0  
        Language  0  
        dtype: int64
```

#### 3.2.1.2 Stop Words

This is an additional data preprocessing tool that utilized for this study so, it has a few stop words that are frequently used in every situation this indicate unnecessary for during data analysis and data extraction because no information and meaning gain in main words thus excluding this stop word from our corpus because it is distorting the models' detection and recognition capabilities and lowering model performance.

### 3.2.1.3 Normalization

For our corpus, employ these data preprocessing methods Lower case text data conversion: Python offers the lower () Function for transforming text data because most of the machines have difficulty comprehending text, for example one, ONE and One this word is the same or one meaning but machine it read or understand as three different words or text because of this reason convert all data set in a lower case format [\[23\]](#).

```
df['Text']=df['Text'].str.lower()
```

### 3.2.1.4 Removal of Punctuation Mark

Most of the time in natural processing NLP punctuation marks is often considered as or call it noisy or irrelevant information[24]. So from our dataset there are many punctuation mark by this removing punctuation can help simply the text or input target can improve the accuracy of models punctuation marks such as periods, commas question marks! " # \$ % & ' ( ) \* + , - . / : ; <= > ? @ [\] ^ \_ ` {||} ~etc can also interfere with certain text analysis techniques such as sentiment analysis or topic modeling .By this selected NLTK tools eliminates punctuation when there is a substantial amount of textual data [\[13\]](#). There for NLTK Eliminating punctuation marks are crucial tools in python.

## 3.3 Tokenization

### 3.3.1 Word Tokenization

Tokenization is dividing the text into individual words or tokens, this step is crucial for building language models as it defines the basic units of our targeted four languages Hadiyya, Wolaytta/Wolayteгна, Somali & Sidama for analysis.



```
Python Edit 🔗  
1 # Calculate the number of word tokens per language  
2 word_tokens_per_language = df.groupby('Language')['Text'].apply(lambda x: x.str.split()).apply(  
3 word_tokens_per_language
```

Table 3.1 Word tokens and average Word Length in our Corpora

Languages	Number of Words/Tokens	Number of unique words	Average Word Length
Hadiyya	27,805	10,049	8.73
Sidama	39,461	16,007	7.91
Somali	66,954	10,108	5.61
Wolayta	35,351	10,107	7.38

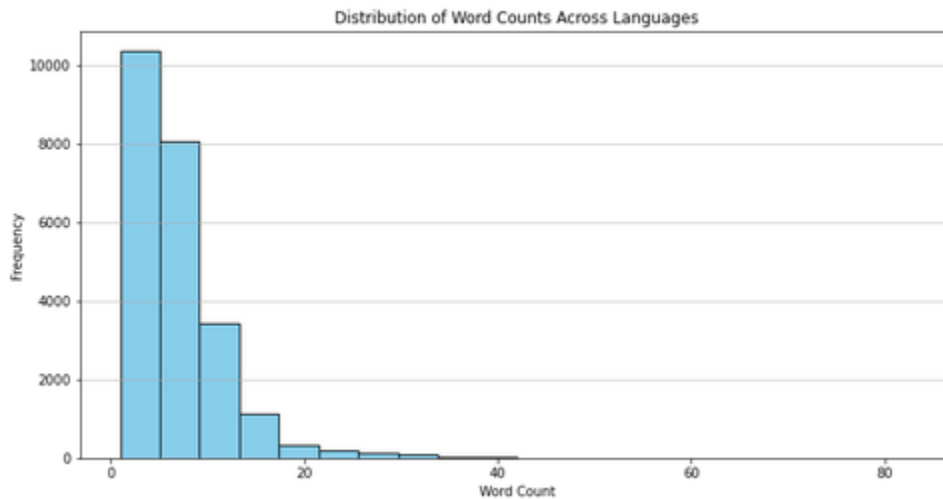


Figure 3.2 Distributions of word Counts across Language

### 3.3.2 Character Tokenization

Character tokenization allows the analysis of character-level patterns and statistical properties. By breaking down the text into its constituent characters, it becomes feasible to extract features that are indicative of a particular language of Hadiyya, Wolaytta/Wolayteгна, Somali&Sidama.

```

1 import pandas as pd
2
3 # Load the data
4 df = pd.read_csv('Languages1.csv', encoding='iso-8859-1')
5
6 # Calculate the number of character tokens per language
7 character_tokens_per_language = df.groupby('Language')['Text'].apply(lambda x: x.str.len().sum)
8 character_tokens_per_language

```

Table 3.2 Character tokens and number of unique words in our Corpora

Languages	Number of characters /Tokens	Number of unique words	Average Word Length
Hadiyya	251,463	10,049	8.73
Sidama	333,887	16,007	7.91
Somali	430,807	10,108	5.61
Wolayta	283,027	10,107	7.38

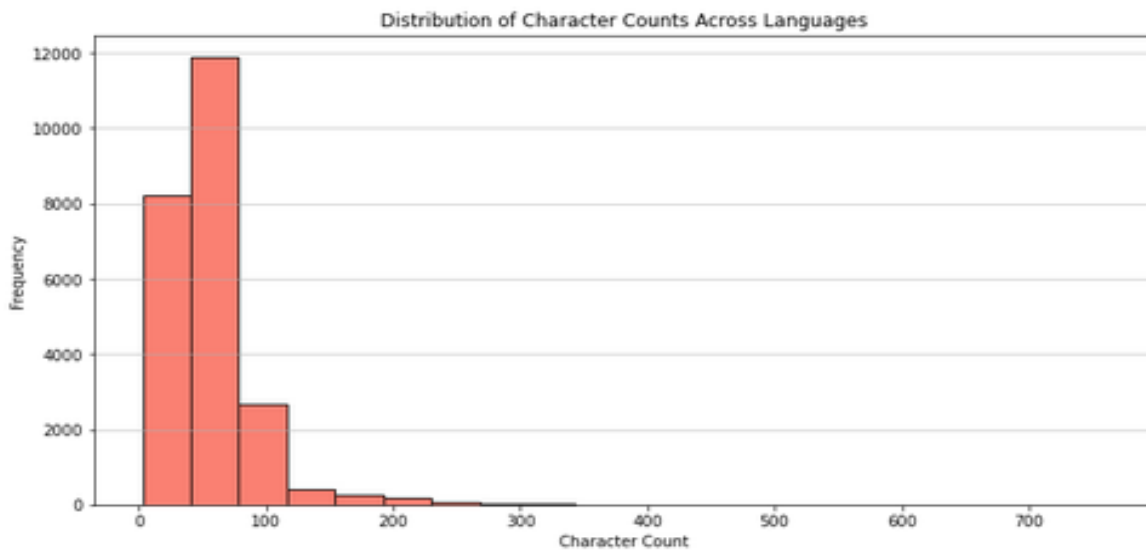


Figure 3.3 Distributions of Character Counts across Language

### 3.4 Stemming

We employed this method to remove affixes in order to derive the root forms or primary words from a text. This process results in normalized text, making it easier for further preprocessing. Stemming is widely applied in tasks such as information retrieval and text mining, playing a crucial role in text preparation [25]. In this research, we created a specialized function for text stemming.

### 3.5 Feature Extraction

We will discuss the extraction of different feature sets that can be used for text language identification. These features provide valuable information about the characteristics of a given text, aiding in language identification. In different studies; n-gram in a character base technique is successful in LID than word level technique. It is easier to create an N-gram feature set for any given text, allowing for a balance between precision and complexity and has been proven achieving success in text language identification and we have been used TF-IDF for converting our data in to a vector form and BoW Counter vector to know vocabulary of words from data and to determine if each data sample contains any Known word[27].

#### 3.5.1 Character N-gram Extraction

Character-based approach focus on the frequencies of individual un-gram or letter combinations (bigrams) in a text and (Char) Analyzer (1,3) were used in this study are not going word by word here we are going character by character extraction.

##### 3.5.1.1 Character Frequencies

Letter frequencies provide insights into the linguistic distribution of characters in different languages. By comparing the frequencies of letters in a text with known frequencies for various languages, we can make informed predictions about the language of the text.

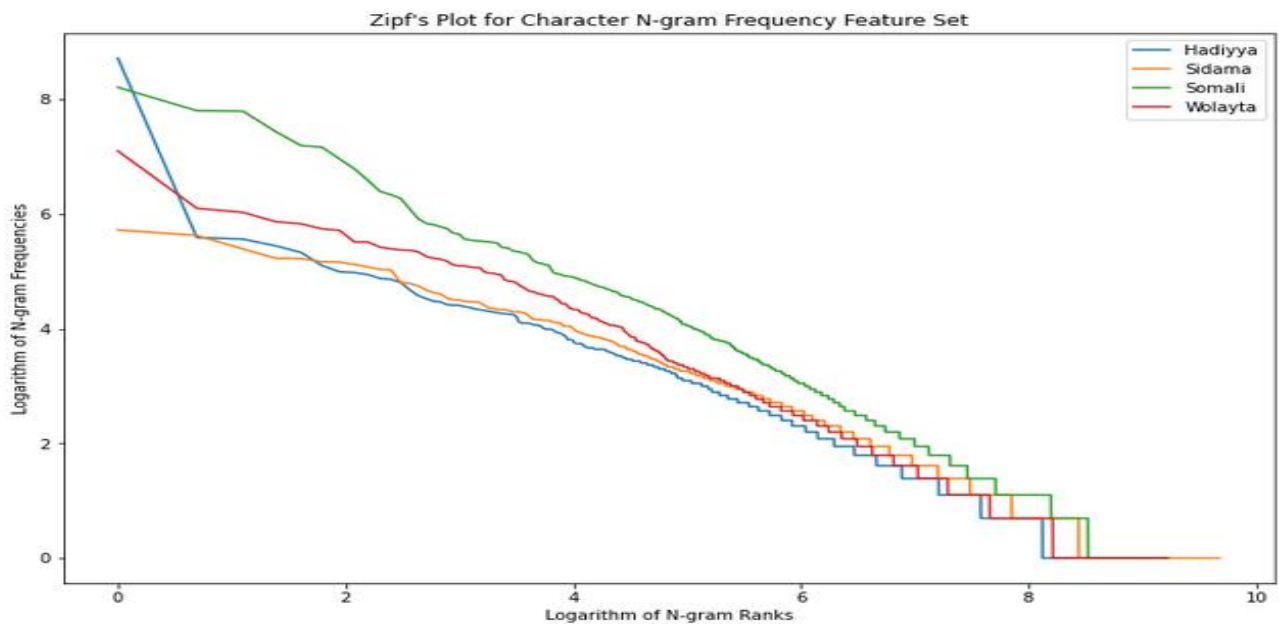


Figure 3.4 Zipf's plots analysis for Character N-gram Frequency per language

```

1 import pandas as pd
2 from collections import Counter
3
4 # Load the data
5 df = pd.read_csv('Languages1.csv', encoding='iso-8859-1')
6
7 # Calculate the top 5 character frequency in each language
8 char_freq = {}
9 for language in df['Language'].unique():
10     text = ''.join(df[df['Language'] == language]['Text'])
11     char_freq[language] = dict(Counter(text).most_common(5))
12
13 char_freq

```

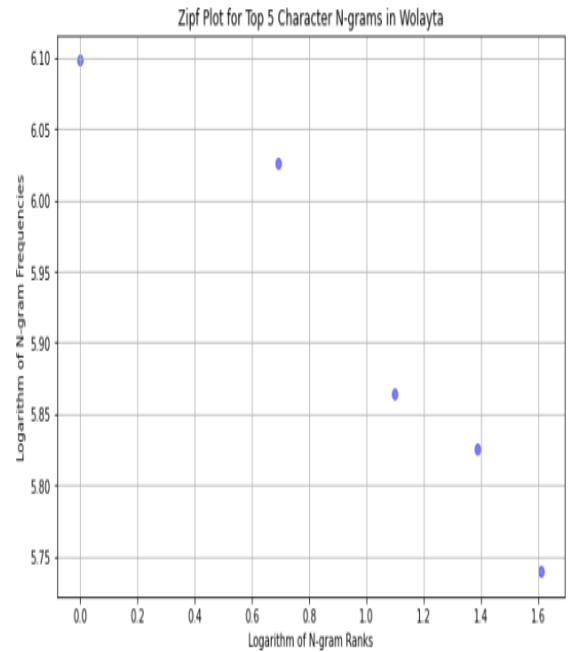
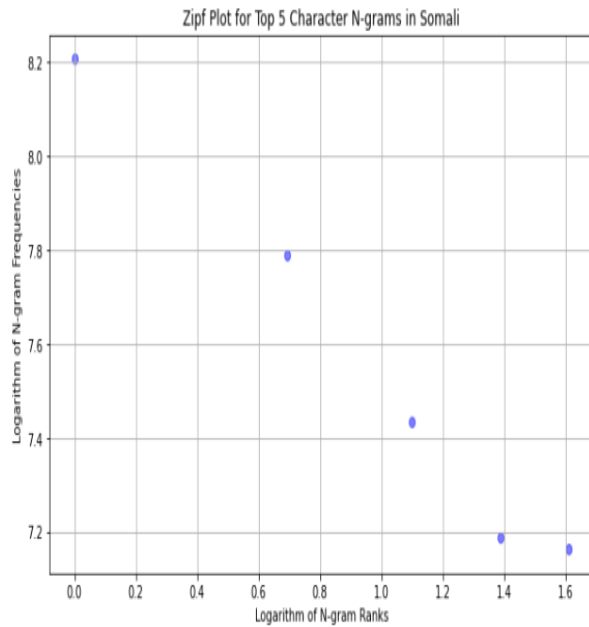
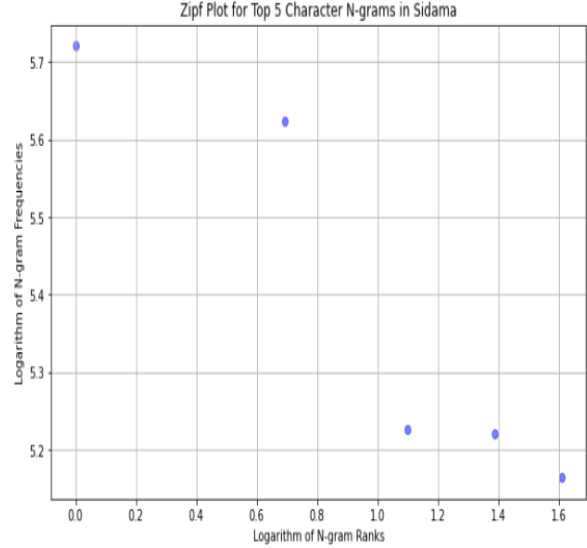
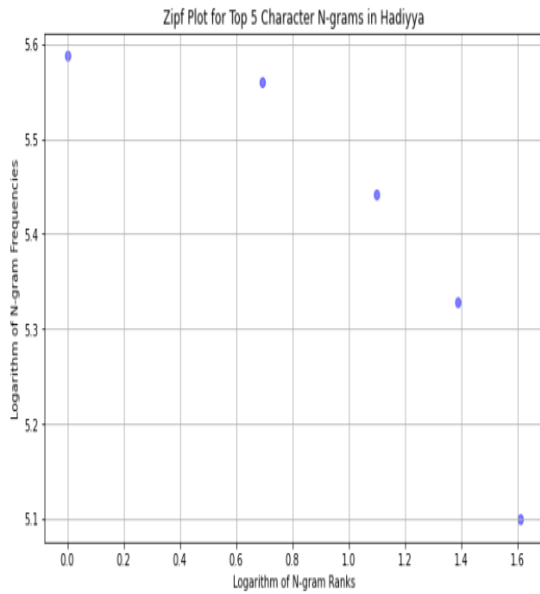


Figure 3.5 Zipf's plots analysis for the top 5 character N-gram per language

### 3.5.1.2 Unigrams and Bigrams Data Extraction

This section demonstrates the utilization of three distinct feature sets in the chosen model; take a Unigrams, Bigrams and Mixture of the most common Unigram & Bigram for this study uses the same dataset.

For instance in my studies experimented with Unigrams, Bigrams and Mixture of both first we'll convert each text into vector using the CountVectorizer, by counting the occurrence of each unique feature and this feature is defined based on n-grams: Unigrams, Bigrams and mixture of both. it's a particular case of n-grams, where n=1, which means one character after creating a dictionary of unique letters (unigrams) by CountVectorizer, it calculates the score of each of them, and based on this, it selects the most important features of each text. In my case our corpus is composed of 87 different features, represented in this list: Number of unigrams in training set: 87 Then To determine the most relevant chars per language, we define the following functions.

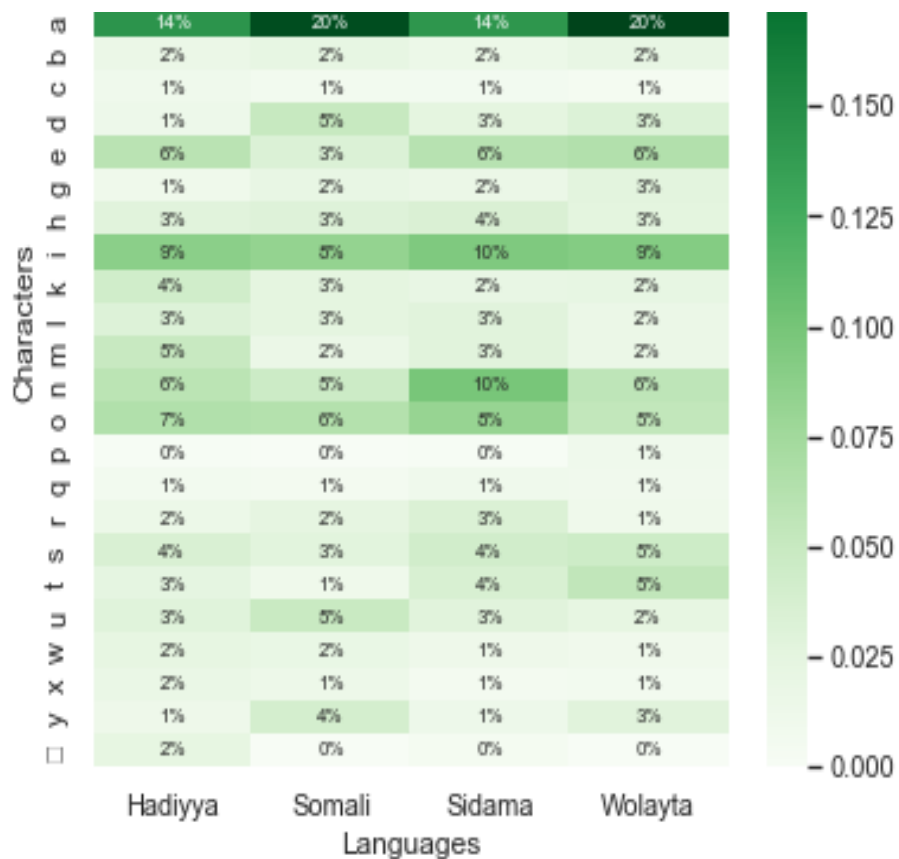


Figure 3.6 Most Common Unigram Chars per Language



From the above Fig 3.6 result using un-gram won't be use full because all languages use almost the same letters. The second feature a particular case of n-grams, where n=2, which means two character after creating a dictionary of unique letters (bigrams) by CountVectorizer, it calculates the score of each of them, and based on this, it selects the most important features of each text. In my case the corpus is composed of 1,815 different features, represented in this list: Number of bigrams in training set: 1,815 .Bi-grams helped us to increase the number of features used in our corpus from 87 in un-gram to 1,815 features listed below. Then To determine the most relevant chars per language, we define bi-gram functions:

```
# Print number of bigrams per language
for lang in languages:
    print(lang, len(relevantCharsPerLanguage[lang]))

Hadiyya 456
Somali 341
Sidama 376
Wolayta 319
```

In Figure 3.6 in one Character extraction we cannot get detail information about the language,so plot heat map of two character extraction and see the result Figure3.7 shows bigram of a text per language from our corpus.

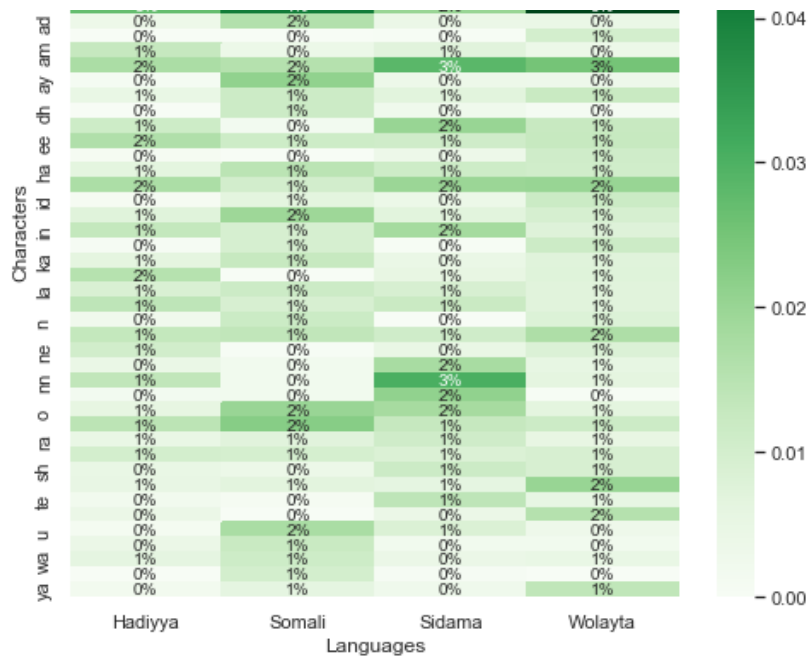


Figure 3.7 Most Common Bigram Chars per Language

As we can see the above Fig3.7 even if it's better that unigrams but we always have some features that are the same in the 4 languages. However our objective is to distinguish between 4 languages. Alternatively, we could also use a mixture of Unigram & Bigram to have more features gets.

### 3.5.2 Word N-gram Extraction

Word-based approach involve based on analyzing the frequencies of words in the text document. In this experiment build a pipeline in a model N-gram range (1-3) words collects one word, one two word, one two three words of a text and analyze statistical measures related to words in a data.

#### 3.5.2.1 Word Frequencies

Word frequencies provide valuable information about the prevalence of specific words in different languages. By comparing the frequencies of words in a given text with those of known languages, we can make informed guesses about the language. Here below shows distribution of word frequency across four languages Hadiyya, Wolaytta/Wolayteгна, Somali & Sidama.

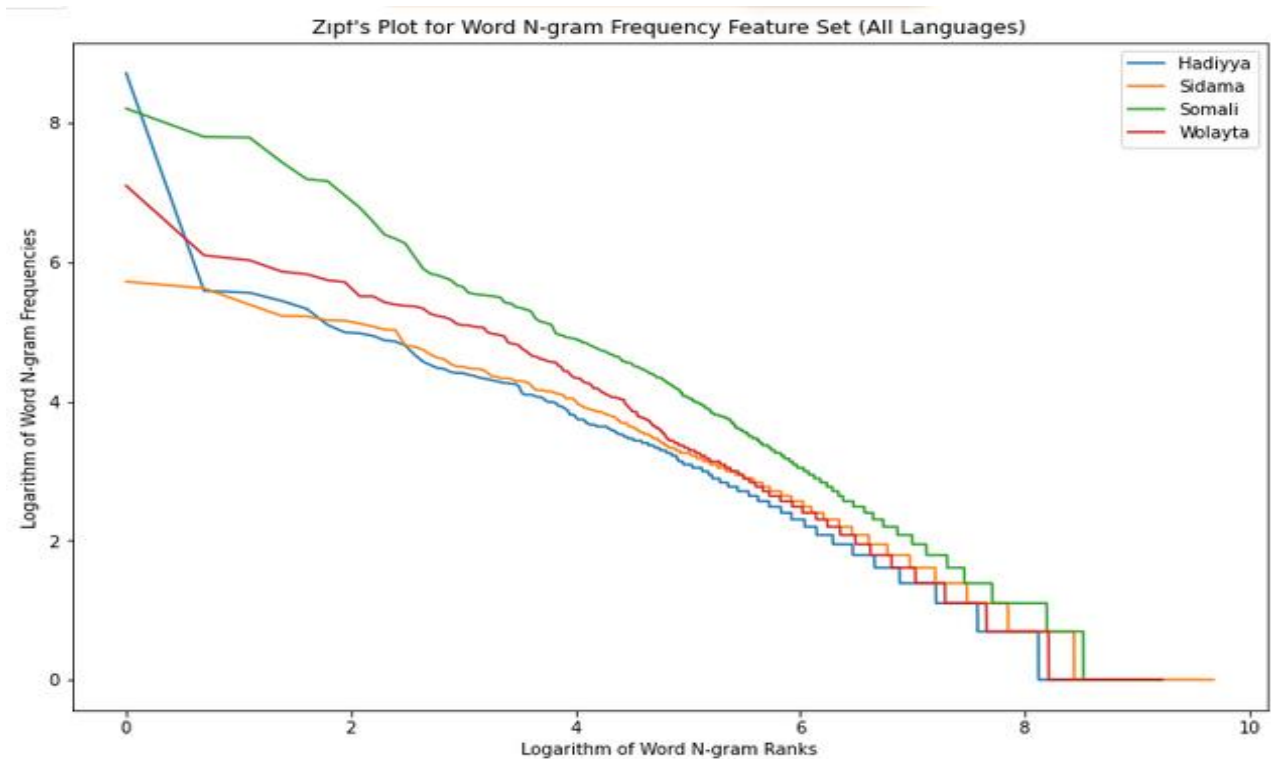


Figure 3.8 Zipf's plots analysis for Word N-gram Frequency per language

### 3.5.2.2 Term Frequency-Inverse Document Frequency (TF-IDF)

TF-IDF measures the importance of a word in a text document by considering both its frequency in the document and its rarity in the entire corpus of documents. [28] [29] This feature can be useful for text language identification as it captures the significance of certain words in different languages. In the field of natural language processing, TF-IDF is commonly used in conjunction with N-gram features. By tokenizing with word-level TF-IDF, tokenizing with character n-gram TF-IDF in my case, initializing this two advanced vectorizer then applying it to the independent features, and finally getting the model to train and fit.

### 3.6 Training and Testing Data

Training of Classifiers is by taking varying proportions of text; used in three languages in Cushitic family and one language in Omotic family. The training data should be utilized for model parameter estimation and the testing data assists in validating the model's performance.

Before train our model first, create the features based on the 'Text' column on all the data set next, splitting our data in two one is for training other is for testing and prepare 23,000 manually tagged records and divide it into 80% (19,040 records) training and 20% (4,761 records) for testing. Table 3.3 demonstrates the total corpus size used to test and train model.

```
# Splitting data into train and test sets
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size = 0.2)
print('X train size: ', len(x_train))
print('X test size: ', len(x_test))
print('y train size: ', len(y_train))
print('y test size: ', len(y_test))

X train size: 19040
X test size: 4761
y train size: 19040
y test size: 4761
```

Table 3.3 Shows Total Corpus size used to Test and Train model

For training (80%)	For testing (20%)
19,040	4,761
Total	23,801

Table 3.4 Show corpus size in sentence level Test and Train model per Language.

```
In [12]: y_train.value_counts()
Out[12]: Hadiyya    5163
         Somali     4823
         Sidama     4612
         Wolayta   4442
         Name: Language, dtype: int64

In [156]: y_test.value_counts()
Out[156]: Hadiyya    1312
          Somali     1228
          Sidama     1137
          Wolayta   1084
          Name: Language, dtype: int64
```

Language	Corpus Size in sentence level	
	For training (80%)	For testing (20%)
Hadiyya	5,163	1,312
Somali	4,823	1,228
Sidama	4,612	1,137
Wolayta	4,442	1,084
Total	19,040	4,761

Table 3.5 Showcorpus size in word level Test and Train model per Language.

Language	Corpus Size in word level		
	Total number of Words/Tokens	For training (80%) Number of Words/Tokens	For testing (20%) Number of Words/Tokens
Hadiyya	27,805	22,244	5,561
Sidama	39,461	31,568	7,892
Somali	66,954	53,563	13,390
Wolayta	35,351	28,280	7,070
Total	169,571	135,656	33,913

## Chapter Four

### 4 Design of Language Identification and Classification Algorithm

#### 4.1 Design of Language Identification

The architectural design of this study in text language identification system consists of three main components: preprocessing, feature extraction, and classification it refers to the construction and framework of systems that can accurately determine the language of a given input text. This process involves analyzing various linguistic features and patterns, such as word frequencies, and character frequency, to classify the language correctly. The design of these language identification systems requires careful consideration of different factors, including the selection of relevant language resources, the choice of feature extraction techniques, and the application of supervised and unsupervised machine learning algorithms. Utilizing these components effectively, language identification systems can achieve high accuracy rates and provide valuable insights in various domains, such as translation services, content filtering, and speech recognition technology. With the constant advancements in computational linguistics and machine learning, the architectural design of language identification continues to evolve, allowing for more accurate and efficient language analysis. This field plays a crucial role in enabling efficient communication, facilitating multilingual interactions, and enhancing overall language processing capabilities[30] [27].

Before identifying the language of a text, first you have to do data preprocessing step like changing all characters in lowercases and converts non-letter characters into white spaces then you have to feed on your language models.

It was not practical to test every potential combination due to many algorithms that are employed in LID. Consequently, document classification methods selected based on their documented effectiveness in studies published in academic journals. In many of the studies, the NB classifier was used the classifier achieves good results in many studies and experiments [31] [32].

For character and word extraction in our case, utilize Unigram (n=1), 1% mixture of Unigram & Bigram, and 60% mixture of Unigram & Bigram in order to identify the text language the second with n-gram range= (1, 3) and analyzer='char' as well By designing a column called 20 (twenty

feature set) to compare text language identification for the research here below shows design of text language identification.

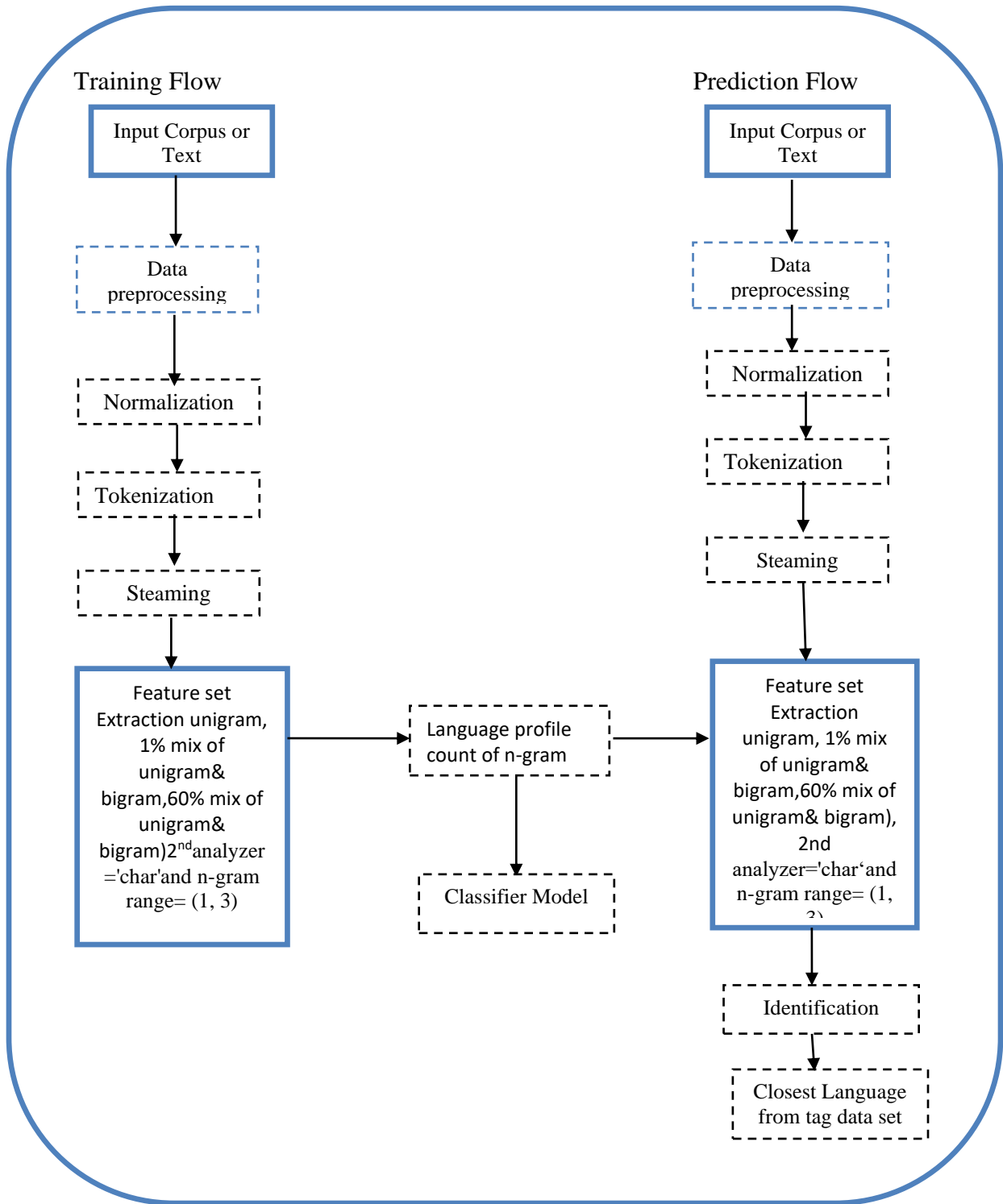


Figure 4.1 General Design of Language Identification Flow

## 4.2 Classification Methods

Classification Methods in NLP are machine learning techniques used to classify text into predefined categories or classes. These algorithms learn patterns and features from labeled data and apply them to unclassified text to assign appropriate labels or classes. In text language identification, classification algorithms play a crucial role in determining the language of a given text by analyzing its linguistic features[33] [34].

Our proposed hybrid approach for Ethiopian language identification combines both supervised and unsupervised learning techniques. These include Naïve Bayes is, Logistic Regression, Random forest, Decision Tree , Gradient Boosting classifiers using statistical approaches, and machine learning algorithms, statistical approaches involve the use of linguistic patterns such as word frequencies, and character frequency and utilization of word based approach and character based approach. Machine learning algorithms learn from large datasets to predict and classify languages accurately. These methods and algorithms play a crucial role in the accurate identification of Ethiopian languages from text data.

## 4.3 Algorithms: Extract Character n-gram and Word n-gram

### 4.3.1 TRAINING: Algorithm: Language Profile Generator

The Language Profile Generator is responsible for creating language profiles. A language profile is a collection of features that represent a particular language. These features can include frequency distributions of characters, words, or phonemes, as well as statistical information such as average word length or sentence length. The generator analyzes a large corpus of text in a specific language and extracts these features to create a language profile.

The Classifier component is responsible for identifying the language of a given text. It uses the language profiles generated by the language profile generator to compare the features of the unknown text with the known language profiles. The classifier calculates the similarity between the features of the unknown text and the language profiles, and assigns a probability or confidence score to each language based on the similarities. The language with the highest probability or confidence score is then determined to be the identified language of the text. The Language Profile Generator and the Classifier work together in a pipeline-like process. The generator creates language profiles for a set of known languages, and these profiles are then used

by the classifier to identify the language of unknown texts. The accuracy of the language identification system depends on the quality and representativeness of the language profiles generated by the profile generator, as well as the effectiveness of the classifier in comparing and matching the features of the unknown text with the language profiles.

#### 4.3.2 TESTING Algorithm: language identification

The testing algorithm for language identification involves multiple steps aimed at accurately determining the language of a given text by calculating the similarity score between the test text and each language profile. Compare the similarity scores and identify the language with the highest score.

#### TRAINING ALGORITHM: LANGUAGE PROFILE GENERATION IN UNIGRAM

Input: Language corpus of Hadiyya, Wolaytta/ Wolayteгна, Somali & Sidama.

Output: Unigram of these languages



- i. Collect training text from our four targeted language.
- ii. Preprocess the text by removing any special characters or symbols.
- iii. Initialize an empty dictionary to store the unigrams and their frequencies and iterate over each word in the preprocessed text then for each word, iterate over each character.
- iv. Create unigrams of characters (individual characters) and store them in the dictionary.
- v. If the unigram already exists in the dictionary, increments its frequency count if the unigram doesn't exist in the dictionary add it with a frequency count of 1.
- vi. For all words in the preprocessed text and Normalize the frequencies by dividing each frequency by the total number of unigrams.
- vii. Sort the unigrams in descending order based on their frequencies.
- viii. Return the top unigrams from the dictionary to represent the language profile.



## TESTING ALGORITHM: LANGUAGE IDENTIFICATION IN UNIGRAM

Input: Text in Hadiyya, Wolaytta/ Wolayteгна, Somali &Sidama.

Output: Closest Language identified from tag data set.

- a. Collect a test text sample for language identification.
- b. Preprocess the test text by removing any special characters or symbols and normalize the text if necessary.
- c. Generate the language profiles for each language using the language profile generation algorithm based on unigrams.
- d. Calculate the similarity score between the test text and each language profile.
- e. Compare the similarity scores and identify the language with the highest score as the predicted language and return the predicted language as the result of the language identification.

## TESTING ALGORITHM: LANGUAGE IDENTIFICATION IN TOP 1% MIXTURE OF UNIGRAM AND BIGRAM

Input: Text in Hadiyya, Wolaytta/ Wolayteгна, Somali &Sidama.

Output: Closest Language identified from tag data set.

- a. Collect a test text sample for language identification.
- b. Preprocess the test text by removing any special characters or symbols and normalize the text if necessary.
- c. Generate the language profiles for each language using the language profile generation algorithm based on the top 1% mixture of unigrams and bi-grams.
- d. Calculate the similarity score between the test text and each language profile.
- e. Compare the similarity scores and identify the language with the highest score as the predicted language and return the predicted language

## TRAINING ALGORITHM: LANGUAGE PROFILE GENERATION IN TOP 1% MIXTURE OF UNIGRAM AND BIGRAM

Input: Language corpus of Hadiyya, Wolaytta/ Wolayteгна, Somali & Sidama.

Output: TOP 1% mixture of unigram & bi-gram relevant character

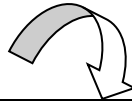


- i. Collect training text from our four targeted language.
- ii. Preprocess the text by removing any special characters or symbols.
- iii. Initialize two empty dictionaries, one for unigrams and one for bi-grams, to store their frequencies.
- iv. Iterate over each word in the preprocessed text and for each word, iterate over each character.
- v. Create unigrams of characters (individual characters) and store them in the unigram dictionary.
- vi. Create bi-grams of characters (pairs of consecutive characters) and store them in the bi-dictionary.
- vii. For both unigrams and bi-grams, consider only the relevant characters for language identification (e.g., excluding spaces or punctuation) and their frequency counts.
- viii. If the n-gram already exists in the corresponding dictionary, increment its frequency count and If the n-gram doesn't exist in the dictionary, add it with a frequency count of 1.
- ix. For all words in the preprocessed text and Normalize the frequencies for both unigrams and bi-grams by dividing each frequency by the total number of relevant n-grams.
- x. Sort the unigrams and bi-grams separately in descending order based on their frequencies and determine the top 1% of the most frequent unigrams and the top 1% of the most frequent bi-grams.
- xi. Combine the selected unigrams and bi-grams to create the language profile for the mixture of unigrams and bi-grams.
- xii. Return the top 1% of the most frequent unigrams & bi-grams of the language profile as the result of the language profile generation algorithm.

## TRAINING ALGORITHM: LANGUAGE PROFILE GENERATION IN TOP 60% MIXTURE OF UNIGRAM AND BIGRAM

Input: Language corpus of Hadiyya, Wolaytta/ Wolayteгна, Somali & Sidama.

Output: Top 60% mixture of unigram & bi-gram relevant character.



- i. Collect training text from our four targeted language.
- ii. Preprocess the text by removing any special characters or symbols.
- iii. Initialize two empty dictionaries, one for unigrams and one for bi-grams, to store their frequencies.
- iv. Iterate over each word in the preprocessed text and for each word, iterate over each character.
- v. Create unigrams of characters (individual characters) and store them in the unigram dictionary.
- vi. Create bi-grams of characters (pairs of consecutive characters) and store them in the bi-dictionary.
- vii. For both unigrams and bi-grams, consider only the relevant characters for language identification (e.g., excluding spaces or punctuation) and their frequency counts.
- viii. If the n-gram already exists in the corresponding dictionary, increment its frequency count and If the n-gram doesn't exist in the dictionary, add it with a frequency count of 1.
- ix. For all words in the preprocessed text and Normalize the frequencies for both unigrams and bi-grams by dividing each frequency by the total number of relevant n-grams.
- x. Sort the unigrams and bi-grams separately in descending order based on their frequencies and determine the top 60% of the most frequent unigrams and the top 60% of the most frequent bi-grams.
- xi. Combine the selected unigrams and bi-grams to create the language profile for the mixture of unigrams and bi-grams.
- xii. Return the language profile as the result of the language profile generation algorithm

## TESTING ALGORITHM: LANGUAGE IDENTIFICATION TOP IN TOP 60% MIXTURE OF UNIGRAM AND BIGRAM

Input: Text in Hadiyya, Wolaytta/ Wolayteгна, Somali & Sidama.



Output: Closest Language identified from tag data set.

- a. Collect a test text sample for language identification.
- b. Preprocess the test text by removing any special characters or symbols and normalize the text if necessary.
- c. Generate the language profiles for each language using the language profile generation algorithm based on the top 60% mixture of unigrams and bi-grams.
- d. Calculate the similarity score between the test text and each language profile.
- e. Compare the similarity scores and identify the language with the highest score as the predicted language.

## TRAINING ALGORITHM: LANGUAGE PROFILE GENERATION CHARACTER ANALYZER(1,3)

Input: Language corpus of Hadiyya, Wolaytta/ Wolayteгна, Somali &Sidama.

Output: character analyzers of lengths from 1 to 3.

- 
- i. Collect training text from our corpus.
  - ii. Preprocess the text by removing any special characters or symbols.
  - iii. Initialize an empty dictionary to store the character analyzer results and their frequencies and Iterate over each character in the preprocessed text.
  - iv. For each character, create character analyzers of lengths from 1 to 3.
  - v. If the character analyzer already exists in the dictionary, increment its frequency count If the character analyzer doesn't exist in the dictionary, add it with a frequency count of 1.
  - vi. For all characters in the preprocessed text and Normalize the frequencies by dividing each frequency by the total number of character analyzers.
  - vii. Sort the character analyzers in descending order based on their frequencies and return the top character analyzers from the dictionary to represent the language profile.
- 

## TESTING ALGORITHM: LANGUAGE IDENTIFICATION CHARACTER ANALYZER(1, 3)

Input: Text in Hadiyya, Wolaytta/ Wolayteгна, Somali &Sidama.

Output: Closest Language identified from tag data set

- a. Collect a test text sample for language identification.
- b. Preprocess the test text by removing any special characters or symbols and normalize the text if necessary.
- c. Generate the language profiles for each language using the language profile generation algorithm based on character analyzers of lengths from 1 to 3.
- d. Calculate the similarity score between the test text and each language profile.
- e. Compare the similarity scores and identify the language with the highest score as the predicted language and return the predicted language as the result of the language identification.

## TRAINING ALGORITHM: LANGUAGE PROFILE GENERATION WORD N-GRAM

RANGE (1, 3)

Input: Language corpus of Hadiyya, Wolaytta/ Wolayteгна, Somali & Sidama.

Output: word n-gram range 1 to 3.



- i. Collect training text from our corpus and preprocess the text by removing any special characters or symbols.
- ii. Initialize an empty dictionary to store the word results and their frequencies and Iterate over each word in the preprocessed text and for each word, create character analyzers of lengths from 1 to 3.
- iii. If the word already exists in the dictionary, increment its frequency count if the word doesn't exist in the dictionary; add it with a frequency count of 1.
- iv. for all word in the preprocessed text and Normalize the frequencies by dividing each frequency by the total number of word
- v. Sort the word in descending order based on their frequencies and return the top word from the dictionary to represent the language profile.



## TESTING ALGORITHM: LANGUAGE IDENTIFICATION WORD N-GRAM

RANGE (1, 3)

Input: Text in Hadiyya, Wolaytta/ Wolayteгна, Somali & Sidama.

Output: Closest Language identified from tag data set

- a. Collect a test text sample for language identification and preprocess the test text by removing any special characters or symbols and normalize the text if necessary.
- b. Generate the language profiles for each language using the language profile generation algorithm based on word n-gram range 1 to 3.
- c. Calculate the similarity score between the test text and each language profile.
- d. Compare the similarity scores and identify the language with the highest score as the predicted language and return the predicted language as the result of the language identification.

# Chapter Five

## 5 Experimental Results

On the chapter demonstrates four targeted language identification experiment, our objective is build a model and identify language within a corpus that has four separate textual language amounts that are related to different languages.

The three types of experiments carried out in this study it is the one was Unigram (n=1), 1% Unigram & Bigram mixture, and 60% Unigram & Bigram mixture another using analyzer='char' and n-gram range= (1, 3) extract character and words and the third experiment By creating 20(twenty feature set) as a column on this each experiment taking different size of testing records from language to language. Each experiment's outcome is examined separately by selected classifier and technique for our targeted language and after reviewing all the experiments will compare each model and shows analysis summary of all result.

As will be detailed below for each model that is chosen, the overall model result, the specific language classification result, and the misclassified records are computed. The tables below display the data statistics for amount of test record which contain all test outcomes and for each of the four languages, different sized documents were utilized.

### 5.1 Experimental Setup

We first create profiles for each of the languages in the dataset, language profile is composed of the frequency counts of unigrams, bigrams, and trigrams of characters in the training data, which are then used as features to train all experiments, follow the same setup process. Figure 5.1 displays the overall framework used to determine steps of the selected algorithm to evaluate.

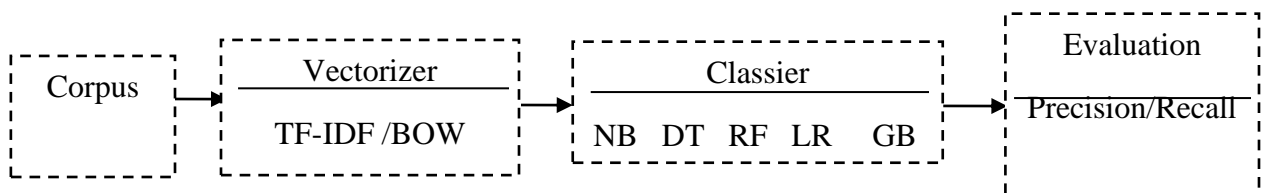


Figure 5.1 Structures of the Experimental Process

Through the use of machine learning system, examine a data, learn from data and make decisions. In this thesis used different tools, software packages are used panda is a power full and quantitative analysis tools ofa python library all data analyses, cleaning, exploring and manipulating data and programming was performed by in python programming language.

Pandas are generally used for data science it is used conjunction with other library like the Numpy library provides high performance multi dimensional array of object and useful linear algebra used as input for plotting function of Matplotlib and statically analysis in scipy and in scikit learn used to implement the algorithm, for data cleaning, data visualizing or final result visualizing.

**Experiment 1:-** The Initial experiment involved examining unigram (n=1), 1% mixture of Unigrams and bigrams, and 60% mixture of unigrams and bigrams

The Initial experiment involved examining unigram (n=1), one percent mixture of unigrams and bigrams, and sixty percent mixture of unigrams and bigrams. This experiment assesses how well a particular feature set can identify languages using Naïve Bayes models. Table 5.1 shows using different selected techniques or features the result was demonstrated.

Using Unigram (n=1) as a feature set, the NB models demonstrated an average result of 81.6%.NB models demonstrated an average result of 95.2% and with feature set consisting of 1% mixture of Unigram & Bigram. Accuracy on average is 91% for a feature set with 60% Unigram & Bigram. Consequently, an average score of 95.2%, the 1% combination of Unigram & Bigram's NB models had the best performance compared to the other feature set.



Table 5.1 Test Result using Naïve Bayes Models

Naïve Bayes Models									
features	Unigram (n=1)			1% combination with(n=1) and (n=2)			60% combination with (n=1) and (n=2)		
Language	amount of Test record	accurately classified	Classification in %	amount of Test record	accurately classified	Classification in %	amount of Test record	accurately classified	Classification in %
Hadiyya	1312	1102	83.9%	1312	1,221	93%	1312	1,176	89.6%
Sidama	1,137	797	70%	1,137	1,055	92.7%	1,137	963	84.6%
Somali	1,228	1,175	95.6%	1,228	1,220	99%	1,228	1,216	99%
Wolayta	1,084	811	74.8%	1,084	1,039	95.8%	1,084	978	90%
Total	4,761	3,885	81.6%	4,761	4,535	95.25%	4,761	4,333	91%

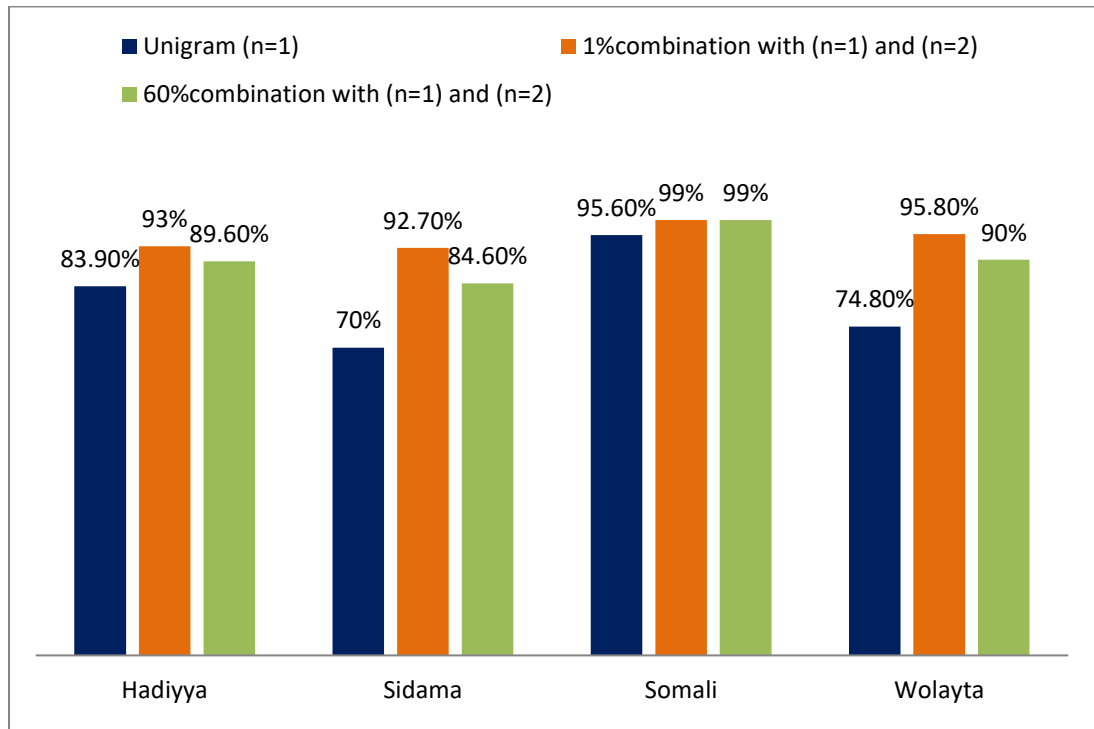


Figure 5.2 Results in NaïveBayes Models

Table 5.2 Test Result using Logistic Regressions Models

Logistic Regressions Models									
features	Unigram(n=1)			1% combination with (n=1) and (n=2)			60% combination with (n=1) and (n=2)		
Language	amount of Test record	accurately classified	Classification in %	amount of Test record	accurately classified	Classification in %	amount of Test record	accurately classified	Classification in %
Hadiyya	1,312	1,117	83.9%	1,312	1,261	96%	1,312	1,220	92.8%
Sidama	1,137	882	77.5%	1,137	1,067	93.8%	1,137	1,019	89.6%
Somali	1,228	1,157	94%	1,228	1,221	99%	1,228	1,213	98.7%
Wolayta	1,084	906	83.5%	1,084	1,055	97%	1,084	1,030	95%
Total	4,761	4,062	85%	4,761	4,604	96.7%	4,761	4,482	94%

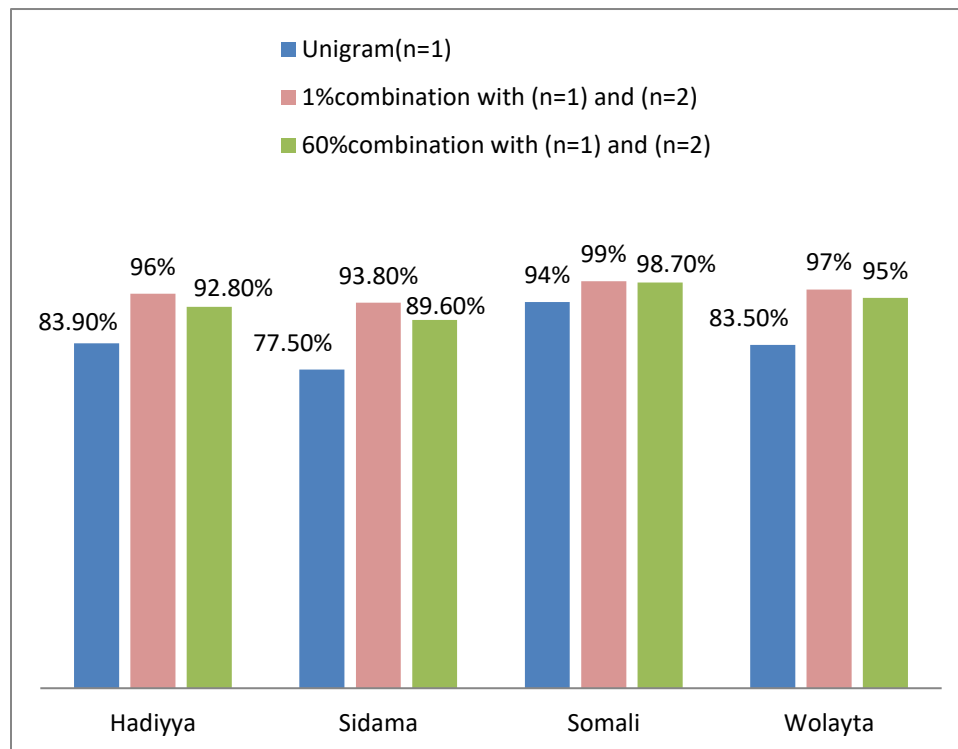


Figure 5.3 Result in Logistic Regressions Models

In Table 5.2, the classifier for logistic regressions when using Unigram (n=1) as a feature set, the average accuracy was 85%.The classifier had an average accuracy of 96.7% in the feature set consisting of 1% mixture of Unigram & Bigram Logistic Regressions. With a 60% Unigram & Bigram feature set mixing, the average accuracy for logistic regressions is 94% therefore; the 1% combination of Unigram & Bigram Naïve Bayes classifier achieved the higher result than others it is average result of 96.7%.

Table 5.3 Test Result using Random Forest Models

Random Forest Models									
Features	Unigram(n=1)			1% combination with (n=1) and (n=2)			60% combination with (n=1) and (n=2)		
Language	amount of Test record	accurately classified	Classification in %	amount of Test record	accurately classified	Classification in %	amount of Test record	accurately classified	Classification in %
Hadiyya	1,312	1,228	93.5%	1,312	1,286	98%	1,312	1,269	96.7%
Sidama	1,137	934	82%	1,137	1,075	94.5%	1,137	1,037	91%
Somali	1,228	1,166	94.9%	1,228	1,216	99%	1,228	1,209	98%
Wolayta	1,084	974	89.8%	1,084	1,068	98.5%	1,084	1,054	97%
Total	4,761	4,302	90.35%	4,761	4,645	97.56%	4,761	4,569	95.96%

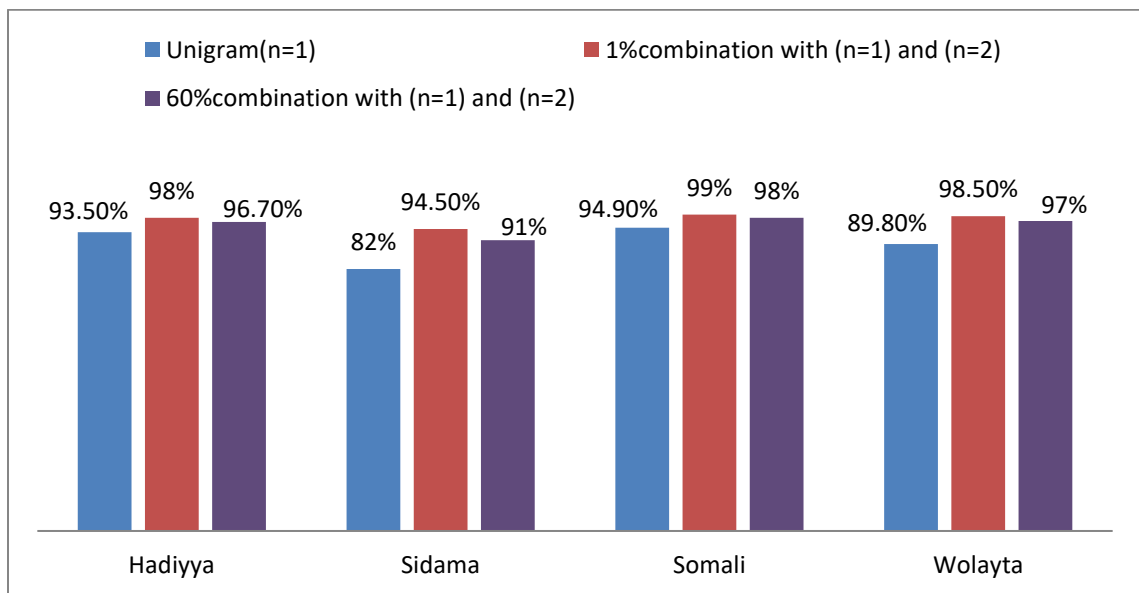


Figure 5.4 Results in Random Forest Models

In Table 5.3, the experiment Random Forest's total performance with the Unigram (n=1) feature set is 90.35%. Overall performance is 97.56% for a 1% mixture of Unigram & Bigram feature set. For 60% mixture of Unigram & Bigram feature set the overall performance is 95.96%. In light of this, performance increase was seen when Random Forest Models were trained using a 1% combination of Unigram & Bigram as the feature set.

Table 5.4 Test Result using Decision Tree Models

Decision Tree Models									
	Unigram(n=1)			1% combination with (n=1) and (n=2)			60% combination with (n=1) and (n=2)		
Language	amount of Test record	accurately classified	Classification in %	amount of Test record	accurately classified	Classification in %	amount of Test record	accurately classified	Classification in %
Hadiyya	1,312	1,106	84%	1,312	1,202	91.6%	1,312	1,167	88.9%
Sidama	1,137	771	67.8%	1,137	991	87%	1,137	936	82%
Somali	1,228	1,089	88.6%	1,228	1,167	95%	1,228	1,137	92.6%
Wolayta	1,084	840	77%	1,084	976	90%	1,084	967	89%
Total	4,761	3,806	79.9%	4,761	4,336	91%	4,761	4,207	88.36%

Unigram (n=1), the decision tree classifier used in Table 5.4, has average result of 79.9% when utilized this technique or feature. The model displayed average result of 91% in the feature set consisting of 1% mixture of Unigram & Bigram Logistic Regressions. For Logistic Regressions a 60% mixture of Unigram & Bigram feature set accuracy on average is 88.36. Therefore, The 1% combination of Uni-Grams and Bi-Grams DT Models outperforms the others with average result of 95.8%. The remaining Unigram (n=1) and 60% combination of Unigram & Bigram feature set, with an average result of 91%.

The experiment presented in Table 5.5 below Gradient Boosting with the Unigram (n=1) feature set performs 89.39% overall. Overall performance is 96.6% for 1% mixture of Unigram & Bigram feature set. The total performance is 94.87% for a 60% Unigram & Bigram feature set mixture. Hence performance improvement observed when 1% mixture of Unigram & Bigram used as a feature set for Gradient Boosting Models. Screenshot of the Gradient Boosting classifier experiment on 1% Unigram & Bigram combination is shown in Figure 5.12.

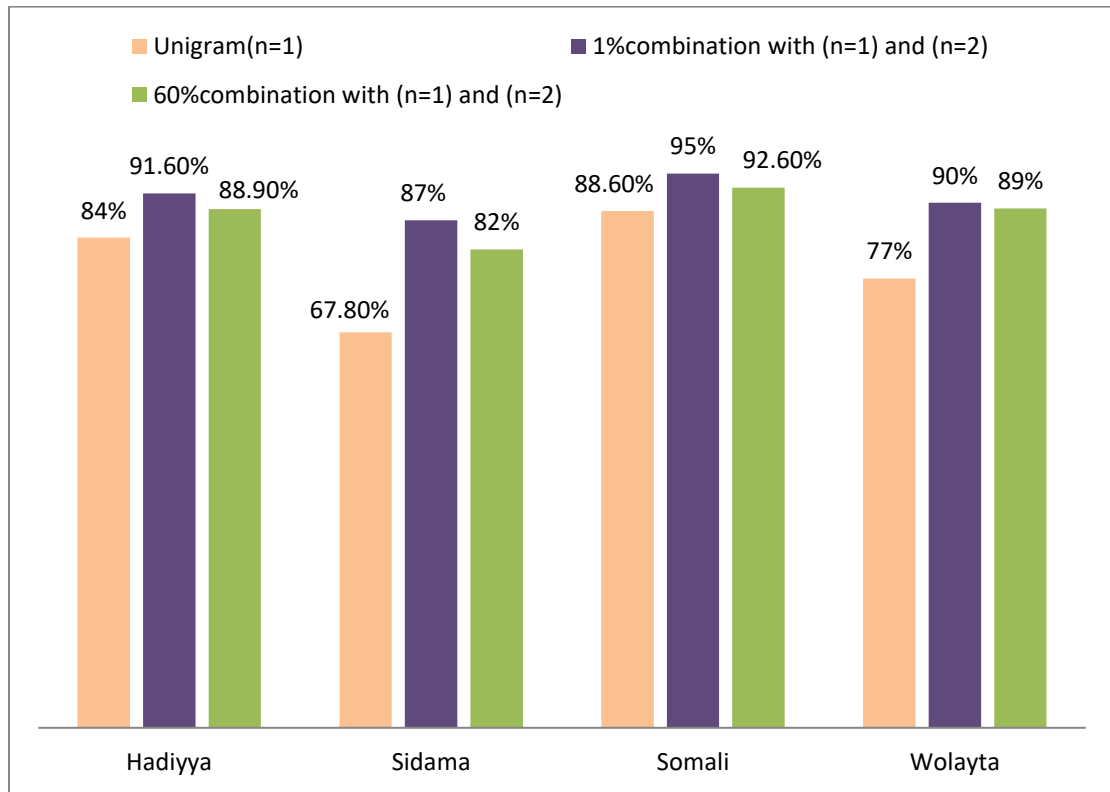


Figure 5.5 Accuracy rates of Decision Tree Models

Table 5.5 Test Result using Gradient Boosting Models

Gradient Boosting Models									
Language	Unigram(n=1)			1% combination with (n=1) and (n=2)			60% combination with (n=1) and (n=2)		
	amount of Test record	accurately classified	Classification in %	amount of Test record	accurately classified	Classification in %	amount of Test record	accurately classified	Classification in %
Hadiyya	1,312	1,195	91%	1,312	1,273	97%	1,312	1,246	94.9%
Sidama	1,137	927	81.5%	1,137	1,054	92.7%	1,137	1,017	89%
Somali	1,228	1,167	95%	1,228	1,212	98.7%	1,228	1,213	98.8%
Wolayta	1,084	967	89%	1,084	1,061	97.9%	1,084	1,041	96%
Total	4,761	4,256	89.39%	4,761	4,600	96.6%	4,761	4,517	94.87%

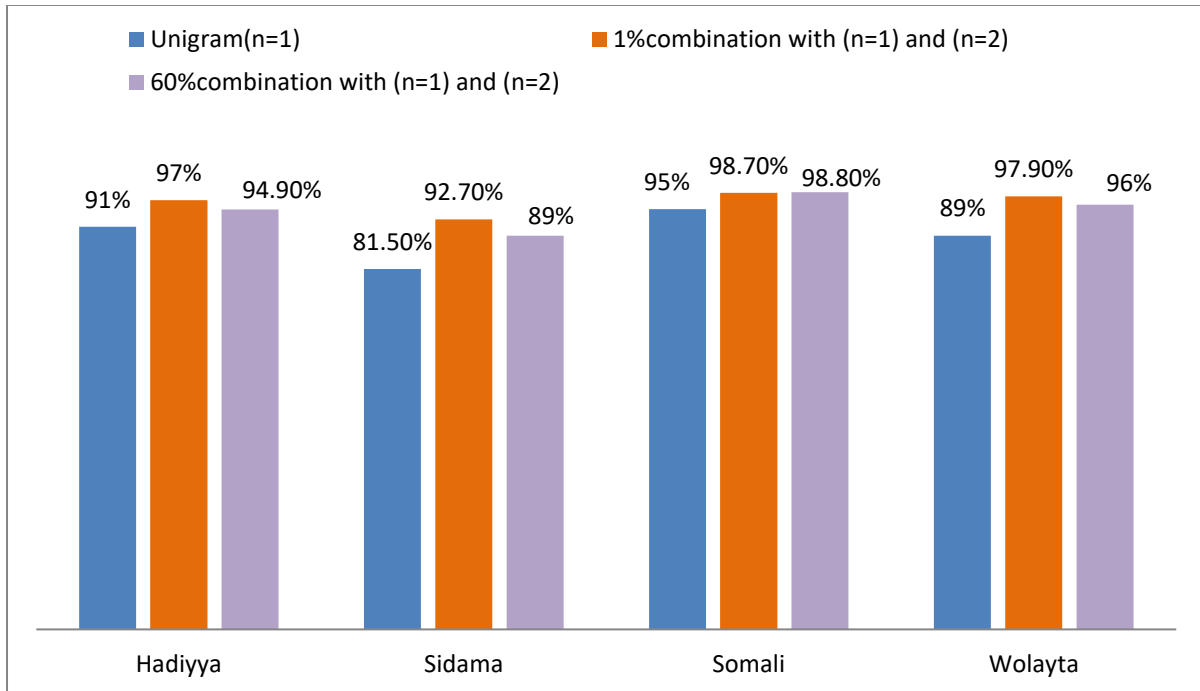


Figure 5.6 Accuracy rates of Gradient Boosting Models

### Experiment 2:- Using N-gram range= (1, 3) and analyzer='char'

In the second experiment is build a pipeline in a model using n-gram range= (1, 3) One, two, and three words are gathered, and then analyzer='char' is used to examine each word character by character. On this case taking different size of testing records from language to language but total test size are equal and the average accuracy result of all language showing below table 5.6.

Table 5.6 Total Test result of using analyzer='Char' and N-gram range= (1, 3)

Classifier	Amount of Test record	Accurately classified	Classification in %
NB Classifier	4,761	4,683	98.36%
LR Classifier	4,761	4,714	98.99%
RF Classifier	4,761	4,690	98.5
DT Classifier	4,761	4,467	93.8%
GB Classifier	4,761	4,634	97.3%

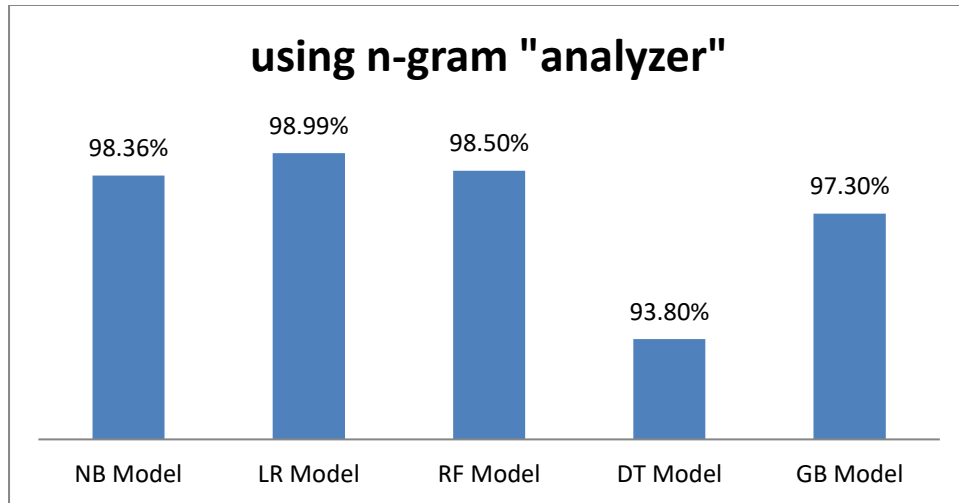


Figure 5.7 Accuracy rates using analyzer='Char' and N-gram range= (1, 3)

Table 5.6 presents the experiment in Logistic regression performs average accuracy 98.9% in overall classifier each language Hadiyya, Sidama, and Somali wolayta is 99%, 98%, 100%, and 99% respectively. The Random Forest model achieved a performance of 98.5%, ranking as the second strongest model in the thus; The Logistic Regression Models demonstrated a peak performance of 98.9%. Figure 5.13 displays a screenshot of the experiment.

### Experiment 3:- Using twenty feature sets are used as column

The third experiment consists of twenty feature sets are used as column, for the first experiment and the second experiment, the third experiment we use the same dataset but very little different size of testing records from language to language. Figure 5.14 Screenshot of Twenty Feature sets

Table 5.7 Test Result of using Random Forest Classifier

Random Forest Classifier			
Language	Amount of Test record	Accurately classified	Classification in %
Hadiyya	1,331	1,148	86.25%
Sidama	1,178	725	61.5%
Somali	1,169	1,057	90.4%
Wolayta	1,083	789	72.85%
Total	4,761	3,719	78.11%

## 5.2 Evaluation Metrics

### 5.2.1 Confusion Matrix

In this section we demonstrate evaluation measures for some of the models employed, we choose the following assessment measures for this research, precision measurement, recall evaluation measurement, and F1 evaluation measurement. In table 5.8 the highlighted in color, confusion matrixes can be understood by examining it shows truly classified for each language this means a True positive result for example from table 5.8 Hadiyya misclassified as sidama language is(3.7%)Somali and Wolaytaare (1.1%),2% respectively this shows a false positive result For Hadiyya Language and also in table 5.8actual input is the sidama language but it is wrongly classified as Hadiyyais (4%),the actual input is Somali but wrongly classified as Hadiyya(0.08%)and the actual input Wolayta but wrongly marked as Hadiyya in (1.8%)this is an example of false negative result for Hadiyya Language.

#### Experiment 1:- Evaluation of Naïve Bayes classifier

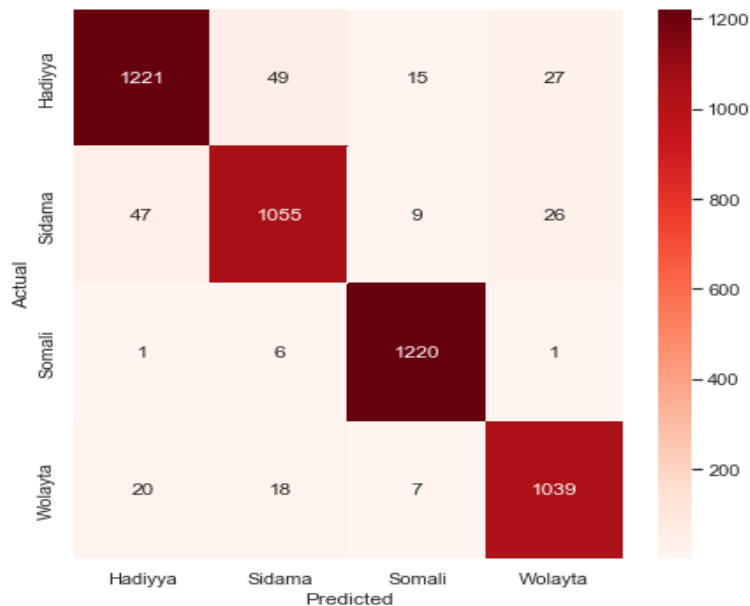


Figure 5.8 Screenshot of NB classifier result on 1% Unigram & Bigram Mixture



Table 5.8 NB classifier Confusion Matrix result on Fig 5.8

	Hadiyya	Sidama	Somali	Wolayta
Hadiyya	1221(93%)	49(3.7%)	15(1.1%)	27(2%)
Sidama	47(4%)	1055(92.7%)	9(0.8%)	26(2.3%)
Somali	1(0.08%)	6(0.5%)	1220(99.3%)	1(0.08%)
Wolayta	20(1.8%)	18(1.7%)	7(0.6%)	1039(95.8%)

Table 5.9 Misclassified Records from Table 5.8

Language	Hadiyya	Sidama	Somali	Wolayta
Misclassified record	91(6.9%)	82(7.2%)	8(0.65%)	45(4%)

From the above confusion matrixes on a Naïve Bayes Classifier, we can see that all errors result from confusions within all four languages. From table 5.9 when using the Unigram & Bigram Mixture for top 1% features Naïve Bayes classifier, the most wrongly classified language is Hadiyya language with an error rate of 6.9%. From table 5.8 Hadiyya language test document was misclassified as Sidama 3.7%, Somali 1.1% and wolayta language 2% of the times. The second most wrongly classified language is Sidama language with an error rate of 7.2 % Sidama language test document was classified as Hadiyya 4% language as Somali 0.8 language as Wolayta language 2.3% of the times.

**Experiment 1:- Evaluation of Logistic Regression classifier**

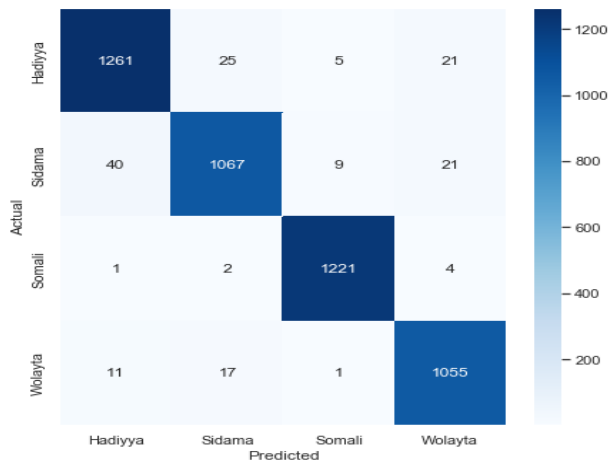


Figure 5.9 Screenshot of Logistic Regression result on 1% Unigram & Bigram Mixture

Table 5.10 Confusion Matrix of Logistic Regression Classifier result on Fig 5.9

	Hadiyya	Sidama	Somali	Wolayta
Hadiyya	1261(96%)	25(1.9%)	5(0.4%)	21(1.6%)
Sidama	40(3.5%)	1067(93.8%)	9(0.8%)	21(1.8%)
Somali	1(0.08%)	2(0.16%)	1221(99.4%)	4(0.3%)
Wolayta	11(1%)	17(1.6%)	1(0.09%)	1055(97%)

Table 5.11 Misclassified Records from Table 5.10

Language	Hadiyya	Sidama	Somali	Wolayta
Misclassified record	51(3.9%)	70(6%)	7(0.6%)	35(3%)

From table 5.11 the Experiment using the Logistic Regression classifier in Unigram & Bigram Mixture for top 1% features the most wrongly classified language is Sidama language with an error rate of 6%. From table 5.10 Sidama language test document was classified as Hadiyya 3.5% and as Somali language 0.8% and Wolayta 1.8% of the times. The second most wrongly classified language is Hadiyya language with an error rate of in total 3.9%. Hadiyya language test document was classified as Sidama language 1.9% of the times, as Somali language 0.4% of the times and as Wolayta language 1.6% of the times.

**Experiment 1:- Evaluation of Random Forest classifier**

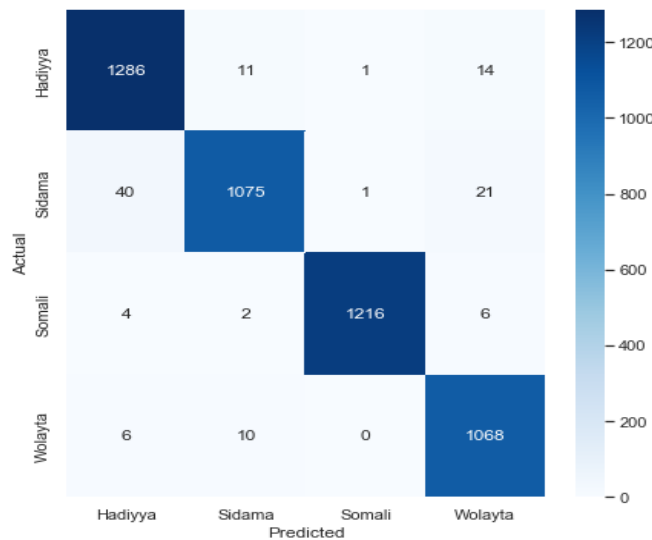


Figure 5.10 Screenshot of RandomForest result on 1% Unigram & Bigram Mixture

Table 5.12 Confusion Matrix of Random Forest Classifier result on Fig 5.10

	Hadiyya	Sidama	Somali	Wolayta
Hadiyya	1286(98%)	11(0.8%)	1(0.07%)	14(1%)
Sidama	40(3.5%)	1075(94.5%)	1(0.08%)	21(1.8%)
Somali	4(0.3%)	2(0.1%)	1216(99%)	6(0.5%)
Wolayta	6(0.6%)	10(0.9%)	0(0%)	1068(98.5%)

Table 5.13 Misclassified Records from Table 5.12

Language	Hadiyya	Sidama	Somali	Wolayta
Misclassified record	26(1.9%)	62(5.5%)	12(0.9%)	16(1.5%)

From table 5.12 and table 5.13 presenting a result using the Random Forest Classifier in Unigram & Bigram Mixture for top 1% features the most wrongly classified language is Sidama language with an error rate of 5.5%. Sidama language test document was classified as Hadiyya 3.5% and as Somali language 0.08% and Wolayta 1.8% of the times. The second most wrongly classified language is Hadiyya language with an error rate of in total 1.9%. Hadiyya language test document was classified as Sidama language 0.8% of the times, as Somali language 0.07 of the times and as Wolayta language 1% of the times.

**Experiment 1:- Evaluation of Decision Tree classifier**

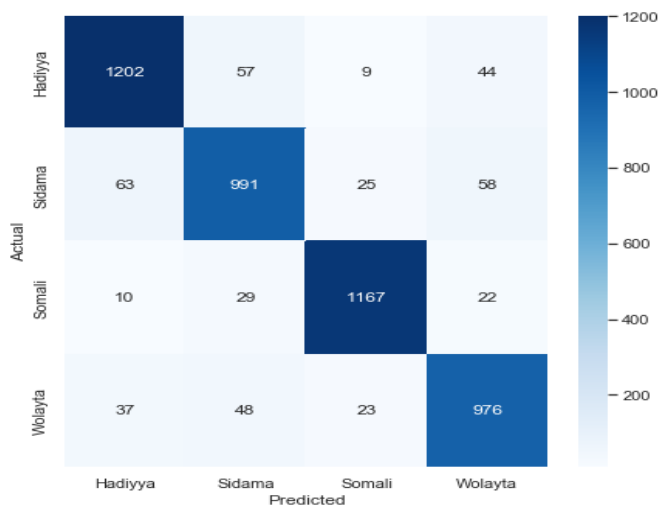


Figure 5.11 Screenshot of Decision Tree result on 1% Unigram & Bigram Mixture

Table 5.14 Confusion Matrix of Decision Tree Classifier result on Fig 5.11

	Hadiyya	Sidama	Somali	Wolayta
Hadiyya	1202 (91.6%)	57(4.3%)	9(0.68%)	44(3.4%)
Sidama	63(5.5%)	991(87%)	25(2%)	58(5%)
Somali	10(0.8%)	29(2.4%)	1167(95%)	22(1.8%)
Wolayta	37(3.4%)	48(4.4%)	23(2.1%)	976(90%)

Table 5.15 Misclassified Records from Table 5.14

Language	Hadiyya	Sidama	Somali	Wolayta
Misclassified record	110(8.4%)	146(12.8%)	61(4.9%)	108(9.9%)

From table 5.15 and table 5.14 Experiment results using the Decision Tree Classifier in Unigram & Bigram Mixture for top 1% features the most wrongly classified language is Sidama language with an error rate of 12.8%. Sidama language test document was classified as Hadiyya 5.5% and as Somali language 2% and Wolayta 5% of the times. The second most wrongly classified language is Hadiyya language with an error rate of in total 8.4%. Hadiyya language test document was classified as Sidama language 4.3% of the times, as Somali language 0.68% of the times and as Wolayta language 3.4% of the times.

### Experiment 1:- Evaluation of Gradient boosting classifier

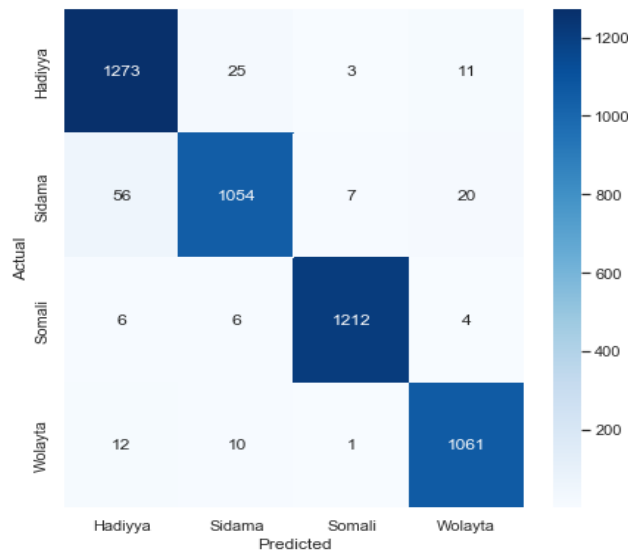


Figure 5.12 Screenshot of Gradient Boosting result on 1% Unigram & Bigram Mixture

Table 5.16 Confusion Matrix of Gradient Boosting Classifier result on Fig 5.12

	Hadiyya	Sidama	Somali	Wolayta
Hadiyya	1273(97%)	25(1.9%)	3(0.2%)	11(0.8%)
Sidama	56(4.9%)	1054(92.7)	7(0.6%)	20(1.8%)
Somali	6(0.5%)	6(0.5%)	1212(98.6%)	4(0.3%)
Wolayta	12(1%)	10(0.9%)	1(0.09%)	1061(97.8%)

Table 5.17 Misclassified Records from Table 5.16

Language	Hadiyya	Sidama	Somali	Wolayta
Misclassified record	39(2.9%)	83(7%)	16(1%)	23(2%)

From table 5.17 the result of the Gradient Boosting Classifier in Unigram & Bigram Mixture for top 1% features shows the most wrongly classified language is Sidama language with an error rate of 7% and also from table 5.16 Sidama language test document was classified as Hadiyya 4.9% and as Somali language 0.6% and Wolayta 1.8% of the times. The second most wrongly classified language is Hadiyya language with an error rate of in total 2.9%. Hadiyya language test document was classified as Sidama language 1.9% of the times, as Somali language 0.2% of the times and as Wolayta language 0.8% of the times.

### **Experiment2:- Evaluation of Logistic Regression classifier**

In the second experiment build a pipeline in to a model and use N-gram (1-3) the system uses by collecting words one word, one two word, and one two three words together with (Char) Analyzer are not going word by word here we are going character by character beside that the first experiment and the second experiment we use the same test dataset but for the third experiment very little difference in the test size data per language.

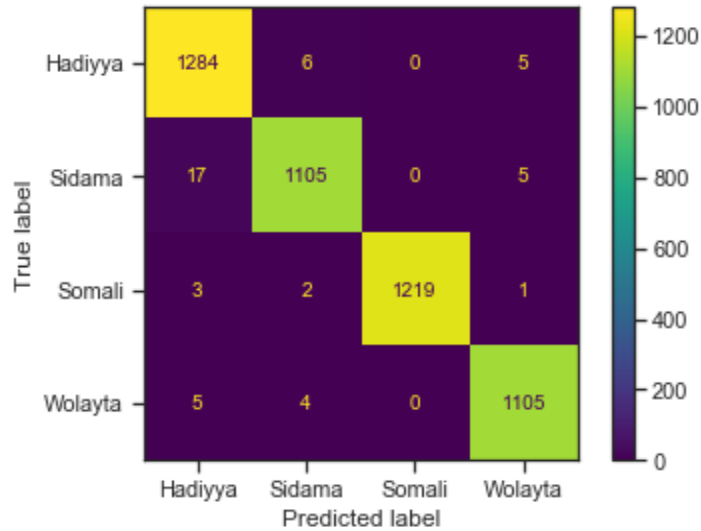


Figure 5.13 Screenshot of Logistic Regression result on analyzer="char" n-gram (1, 3)

Table 5.18 Confusion Matrix of Logistic Regression Classifier result on Fig 5.13

	Hadiyya	Sidama	Somali	Wolayta
Hadiyya	1284(99.1%)	6(0.46%)	0(0%)	5(0.38%)
Sidama	17(1.5%)	1105(98%)	0(0%)	5(0.4%)
Somali	3(0.2%)	2(0.16%)	1219(99.5%)	1(0.08%)
Wolayta	5(0.4%)	4(0.35%)	0(0%)	1105(99%)

Table 5.19 Misclassified Records from Table 5.18

Language	Hadiyya	Sidama	Somali	Wolayta
Misclassified record	11(0.84%)	22(1.95%)	6(0.6%)	6(0.48%)

From table 5.19 the Experiment result using the Logistic Regression classifier using (Char) Analyzer, N-gram (1-3) features the most wrongly classified language is Sidama language with an error rate of 1.95%. The second most wrongly classified language is Hadiyya language with an error rate of in total 0.84%. Hadiyya language test document was wrongly marked as Sidama language 0.46% of the times, as Somali language 0% of the times and as Wolayta language 0.38% of the times.

In the third experiment twenty feature sets are used to train our model, here below the result obtained per language and for the model test data in the third experiment used very little difference per language.

Out[5]:

Language	Hadiyya	Sidama	Somali	Wolayta
<b>word_count</b>	4.305310	6.862783	11.066777	6.378744
<b>character_count</b>	34.686174	52.166783	60.741322	45.472573
<b>word_density</b>	0.117491	0.128883	0.179048	0.136213
<b>punc_count</b>	2.117820	1.180348	1.885785	1.406893
<b>v_char_count</b>	0.003406	0.015478	0.000165	0.000000
<b>w_char_count</b>	0.904320	0.753565	1.367273	0.543486
<b>ij_char_count</b>	0.009135	0.002957	0.005289	0.000541
<b>num_double_consec_vowels</b>	2.032977	2.730957	5.999339	3.316853
<b>num_consec_vowels</b>	0.004954	0.116696	0.001653	0.670696
<b>num_vowels</b>	4.244001	6.729391	11.034711	6.187117
<b>vowel_density</b>	0.988140	0.977264	0.995986	0.969179
<b>capitals</b>	0.720855	1.345739	1.681488	1.997474
<b>caps_vs_length</b>	0.019121	0.028055	0.029702	0.046639
<b>num_exclamation_marks</b>	0.002477	0.032000	0.000661	0.041862
<b>num_question_marks</b>	0.684162	0.083826	0.041488	0.117827
<b>num_punctuation</b>	2.117820	1.180348	1.885785	1.406893
<b>num_unique_words</b>	4.265676	6.724870	10.441653	6.232588
<b>num_repeated_words</b>	0.038861	0.123826	0.515207	0.129737

Figure 5.14 Screenshot of Twenty Feature sets are used as Column

Figure 5.15 an average result of the Random Forest Model for the given test record was 78.11%,so for all classifier best performance was obtained by Random Forest Classifier.

### Experiment 3:- Evaluation of Random Forest classifier

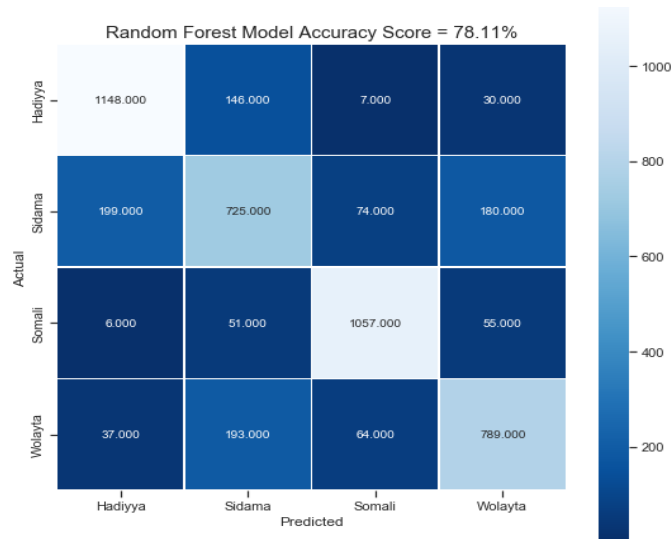


Figure 5.15 Screenshot of Random Forest Classifier Confusion Matrix result

Table 5.20 Random Forest Classifier Result on Fig 5.15

	Hadiyya	Sidama	Somali	Wolayta
Hadiyya	1148(86.25%)	146(10.96%)	7(0.5%)	30(2.25%)
Sidama	199(16.89%)	725(61.5%)	74(6.28%)	180(15.28%)
Somali	6(0.5%)	51(4.36%)	1057(90.4%)	55(4.7%)
Wolayta	37(3.4%)	193(17.8%)	64(5.9%)	789(72.85%)

Table 5.21 Misclassified Records from Table 5.20

Language	Hadiyya	Sidama	Somali	Wolayta
Misclassified record	183(13.74%)	453(38.45%)	112(9.58%)	294(27.14%)

From table 5.21 When using the Random Forest Classifier in twenty feature sets used as column the most wrongly classified language is Sidama language with an error rate of 38.45%. Sidama language test document was classified as Hadiyya 16.89% and as Somali language 6.28% and Wolayta 15.28% of the times and also from table 5.21 the second most wrongly classified language is Hadiyya language with an error rate of in total 13.74%. Hadiyya language test document was classified as Sidama language 10.96% of the times, as Somali language 0.5% of the times and as Wolayta language 2.25% of the times.



### 5.2.2 Accuracy

In my study The Unigram & Bigram Mixture for the top 1% features yielded the best overall results. It is 95% of accuracy in NB, 97% for RF Model, 91% for DT Model, and 96% for LR Models there will be 96% accuracy in GB Models.

### 5.2.3 Precision

This is the first evaluation measurement for the model used in the experimentation In Table 5.22 the highest precision for our model NB classifier in Unigram & Bigram Mixture for top 1% is Somali language 0.98% it is calculated as

$$\text{Precision} = \frac{(\text{True Positive(Tp)})}{(\text{True Positive(Tp)} + \text{False Positive(FP)})}$$

For example in Table 5.81, 220 are predicted positively as Somali language by NB model and also Table 5.8 matrix, 31 times misclassified as negative in Somali language by NB model this is calculated as:

$$\text{Precision} = \frac{1220}{1220+15+9+7} = 0.98 \text{----- (equ 1)}$$

Equation

In table 5.12 over all used Model the highest precision get in Random Forest Classifier Unigram & Bigram Mixture for top 1% features in Somali Language 100% it is calculated as:

$$\text{Precision} = \frac{1216}{1216+1+1} = 0.998 \text{----- (equ 2)}$$

In the Table 5.10 and Table 5.16 shows over all used Model the second highest precision get in Logistic Regression Classifier and Gradient Boosting Unigram & Bigram Mixture for top 1% features in Somali Language it is correct around 0.987 and 0.991% respectively logistic regression it is calculated as:

$$\text{Precision LR} = \frac{1221}{1221+5+9+1} = 0.987 \text{ Precision GB} = \frac{1212}{1212+7+3+1} = 0.991 \text{----- (equ 3)}$$

## Experiment 1:- Evaluation of all classifier

Table 5.22 Precision Result

NB classifier Unigram & Bigram Mixture for top 1%		Logistic Regression Unigram & Bigram Mixture for top 1%		Random Forest Classifier Unigram & Bigram Mixture for top 1% features.		Decision Tree Classifier Unigram & Bigram Mixture for top 1%		Gradient Boosting Classifier Unigram & Bigram Mixture for top 1%	
Targeted Input	Result Precision	Targeted Input	Result	Targeted Input	Result	Targeted Input	Result	Targeted Input	Result
Hadiyya	0.95%	Hadiyya	0.96%	Hadiyya	0.96%	Hadiyya	0.92%	Hadiyya	0.95%
Sidama	0.95%	Sidama	0.96%	Sidama	0.96%	Sidama	0.88%	Sidama	0.96%
Somali	0.98%	Somali	0.99%	Somali	1.00%	Somali	0.95%	Somali	0.99%
Wolayta	0.95%	Wolayta	0.96%	Wolayta	0.96%	Wolayta	0.89%	Wolayta	0.97%
(Ma) average	0.95%	(Ma) average	0.97%	(Ma) average	0.98%	(Ma) average	0.91%	(Ma) average	0.97%
(We) average	0.95%	(We) average	0.97%	(We) average	0.98%	(We) average	0.91%	(We) average	0.97%

### 5.2.4 Recall

This is the second evaluation measurement for the model used in the experimentation, in table 5.23 the Highest Recall for our model NB classifier in Unigram & Bigram Mixture for top 1% is Somali language 0.99% it is calculated as:

$$\text{Recall} = \frac{(\text{True Positive}(Tp))}{(\text{True Positive}(Tp) + \text{False negative}(Fn))}$$

$$\text{Recall} = \frac{1220}{1220+1+6+1} = 0.991 \text{----- (equ 4)}$$

As you know in the case of precision value the denominator of (FP) was by taking the y-axis value which is a misclassified record in Somali language but from table 5.8, Table 5.10 and Table 5.12 for the recall result the denominator of (FN) by taking x-axis value which is wrongly prediction of Somali language.

Over all used model the highest recall get in Logistic Regression Classifier Unigram & Bigram Mixture for top 1% features in Somali language is 0.994%, NB classifier in Unigram & Bigram Mixture for top 1% is Somali language is 0.993 and Gradient Boosting Unigram & Bigram Mixture for top 1% features in Somali Language it is correct around 0.986 it is calculated as follows:

$$\text{Recall for LG} = \frac{1221}{1221+1+2+4} = 0.994 \text{----- (equ 5)}$$

$$\text{Recall for NB} = \frac{1220}{1220+1+6+1} = 0.993 \text{Recall for GB} = \frac{1212}{1212+6+6+4} = 0.986 \text{----- (equ 6)}$$

Table 5.23 Recall Result

NB classifier Unigram & Bigram Mixture for top 1%		Logistic Regression Unigram & Bigram Mixture for top 1%		Random Forest Classifier Unigram & Bigram Mixture for top 1% features		Decision Tree Classifier Unigram & Bigram Mixture for top 1%		Gradient Boosting Classifier Unigram & Bigram Mixture for top 1%	
Targeted Input	Recall Result %	Targeted Input	Result %	Targeted Input	Result %	Targeted Input	Result %	Targeted Input	Result %
Hadiyya	0.93	Hadiyya	0.96	Hadiyya	0.98	Hadiyya	0.92	Hadiyya	0.97
Sidama	0.93	Sidama	0.94	Sidama	0.95	Sidama	0.87	Sidama	0.93
Somali	0.99	Somali	0.99	Somali	0.99	Somali	0.95	Somali	0.99
Wolayta	0.96	Wolayta	0.97	Wolayta	0.99	Wolayta	0.90	Wolayta	0.98
(Ma) average	0.95	(Ma) average	0.97	(Ma) average	0.98	(Ma) average	0.91	(Ma) average	0.97
(We) average	0.95	(We) average	0.97	(We) average	0.98	(We) average	0.91	(We) average	0.97

### 5.2.5 F1 Result

This is the third Evaluation measurement for the model in the experimentation in Table 5.24 the Highest F1 Result for our model NB classifier in Unigram & Bigram Mixture for top 1% is Somali language 0.98% it is calculated as:

$$F1 \text{ score} = \frac{2(\text{Precision} * \text{Recall})}{(\text{Precision} + \text{Recall})}$$

$$F1 \text{ score} = \frac{2 * (0.98 * 0.99)}{(0.98 + 0.99)} = 0.98$$

Over all used model the highest F1 Score get in Random Forest Classifier Unigram & Bigram Mixture for top 1% features in Somali Language 0.994% it is calculated as:

$$F1\ score = \frac{2*(1*0.99)}{1+0.99} = 0.994\text{-----} (equ\ 6)$$

Table 5.24 F1 Score Result

NB classifier Unigram & Bigram Mixture for top 1%		Logistic Regression Unigram & Bigram Mixture for top 1%		Random Forest Classifier Unigram & Bigram Mixture for top 1% features.		Decision Tree Classifier Unigram & Bigram Mixture for top 1%		Gradient Boosting Classifier Unigram & Bigram Mixture for top 1%	
Input	F1 Result %	Input	Result %	Input	Result %	Input	Result %	Input	Result %
Hadiyya	0.94	Hadiyya	0.96	Hadiyya	0.97	Hadiyya	0.92	Hadiyya	0.96
Sidama	0.93	Sidama	0.95	Sidama	0.96	Sidama	0.88	Sidama	0.94
Somali	0.98	Somali	0.99	Somali	0.99	Somali	0.95	Somali	0.99
Wolayta	0.95	Wolayta	0.97	Wolayta	0.97	Wolayta	0.89	Wolayta	0.97
(Ma) average	0.95	(Ma) average	0.97	(Ma) average	0.98	(Ma) average	0.91	(Ma) average	0.97
(We) average	0.95	(We) average	0.97	(We) average	0.98	(We) average	0.91	(We) average	0.97

In table 5.24 over all used model the second highest F1 Result get in Logistic Regression Classifier and Gradient Boosting Unigram & Bigram Mixture for top 1% features in Somali Language in both model it is around 0.989 it is calculated as:

$$F1\ Score_{LR} = \frac{2(0.99*0.99)}{0.99+0.99} = 0.989 \quad F1\ Score_{GB} = \frac{2(0.99*0.99)}{0.99+0.99} = 0.989\text{-----} (equ\ 7)$$

## Experiment2:- Evaluation of Logistic Regression

Logistic Regression Classifier Output				
	precision	recall	f1-score	support
Hadiyya	0.98	0.99	0.99	1295
Sidama	0.99	0.98	0.98	1127
Somali	1.00	1.00	1.00	1225
Wolayta	0.99	0.99	0.99	1114
accuracy			0.99	4761
macro avg	0.99	0.99	0.99	4761
weighted avg	0.99	0.99	0.99	4761

Figure 5.16 Screen Shoot Result using analyzer='Char' and N-gram range= (1, 3)

Second experiment From the above figure 5.16 When using analyzer='char' and n-gram range=(1, 3) are used as feature set from Over all used Model the highest result get in LR Classifier and in particular, Somali language achieved 100% classified.

## Experiment 3:- Evaluation of Random Forest classifier

Random Forest Model Output				
	precision	recall	f1-score	support
Hadiyya	0.83	0.86	0.84	1331
Sidama	0.65	0.62	0.63	1178
Somali	0.88	0.90	0.89	1169
Wolayta	0.75	0.73	0.74	1083
accuracy			0.78	4761
macro avg	0.78	0.78	0.78	4761
weighted avg	0.78	0.78	0.78	4761

Figure 5.17 Screen Shoot Result on Twenty Feature sets

In the third experiment from the above Figure when using twenty feature sets are used as column from Over all used model the highest result get in RF Model in Somali Language it is 0.90%, it is calculated as follows.

$$Recall \text{ for } Rf \text{ Somali} = \frac{1057}{1057+6+51+55} = 0.90 \text{----- (equ 8)}$$

### 5.3 Comparative Analysis

This section demonstrate comparison of different algorithm with different feature set and summary result of each model out performs, in Table 5.25 shows the result of each model being used Unigram (n=1),1% mixture of Unigrams and Bigrams , 60%mixture of Unigrams and Bigrams ,analyzer='char' and n-gram range=(1, 3)and twenty feature sets used as a column.

Table 5.25 Comparison of Different algorithm with Different feature set

supervised learning approached		Result in NB Classifier	Result in LR Classifier	Result in RF Classifier	Result in DT Classifier	Result in GB Classifier
Unsupervised learning/ Statistical approaches	EX 1 Unigram (n=1)	81.6%	85%	90.35%	79.9%	89.39%
	1% mixture of Unigrams and Bigrams	95.25%	96.7%	97.56%	91%	96.6%
	60% mixture of Unigrams and Bigrams	91%	94%	95.96%	88.36%	94.87%
	EX 2 analyzer='char' and n-gram range=(1, 3)	98.36%	98.99%	98.5%	93.8%	97.3%
Individual ML approaches	EX 3 twenty feature sets used as a column	59.71% in (Gaussian Naive Bayes)	70.41%	78.11%	73.18%	76.69%

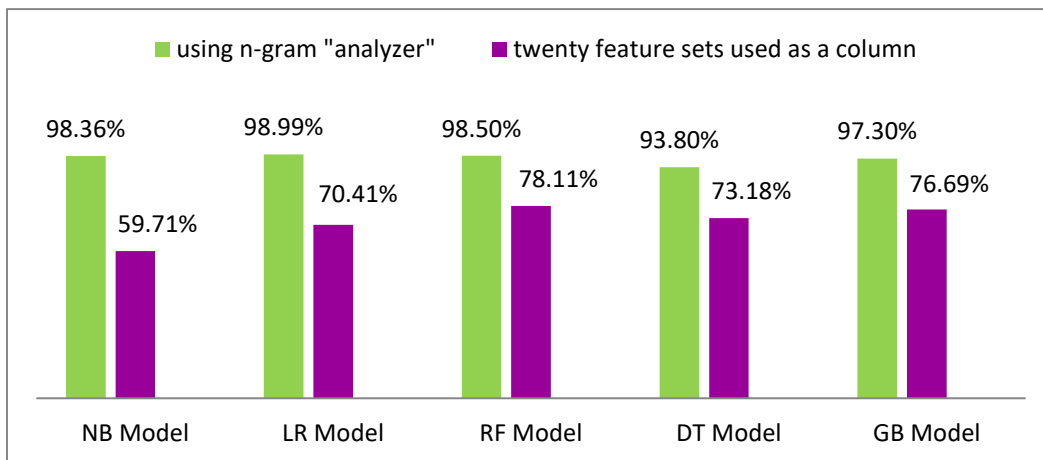
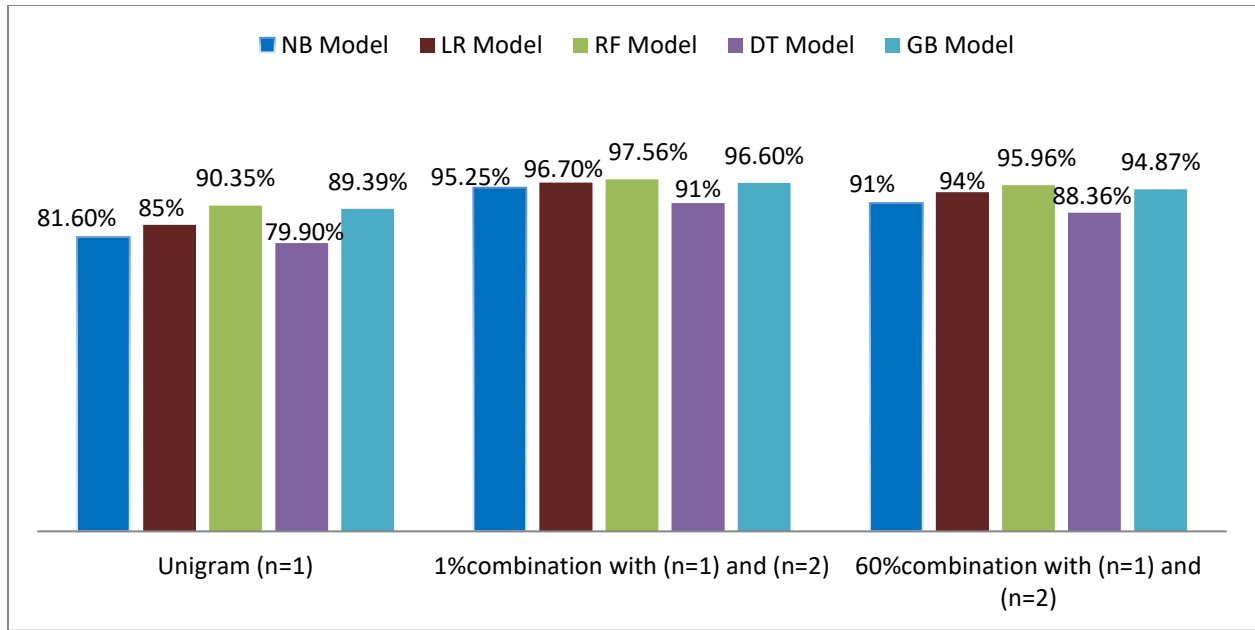


Figure 5.18 Accuracy Comparison of Different Classifier

According to the analysis table the accuracy rates of Logistic Regression in analyzer='char 'and n-gram range= (1, 3) is comparatively better than other model, The Random Forest classification has been successful in case of 1% mixture of Unigrams and Bigrams feature set it is the highest accuracy rates in NB classifier and Gradient Boosting Classifier which uses 1% mixture of Unigrams and Bigrams extracted features it is second in accuracy Result.



## Chapter Six

### 6 Conclusion and Recommendation

#### 6.1 Conclusion

The main focus of this paper is text identification of Low-resourced Ethiopian language of Hadiyya, Sidama, Somali and Wolayta. A method was created to identify targeted language, in a given corpus sources for input text. In this study we select mostly used text classification approach and techniques it is Unigram (n=1), 1% mixture of Unigrams and Bigrams, 60% mixture of Unigrams and Bigrams, analyzer='char' and n-gram range= (1, 3) and twenty feature sets used as column. In the first experiment, for all classifiers, we employed a unigram (n=1) feature set with four specific language instruction classes. In the Naïve Bayes model, the average classification accuracy was 81%, 85%, 90%, 79%, and 89% for LR, RF, DT, and GB classifiers respectively. Average classification accuracy of the NB, LR, and RF, DT, GB classifiers in a 1% mixture of Unigrams and Bigrams was 95.25%, 96.7%, 97.56%, 91%, and 96.6%, respectively. On this to compare the accuracy of the first feature set unigram (n=1) with 1% mixture of Unigrams and Bigrams, the accuracy of the model in 1% mixture of Unigrams and Bigrams shows improved in all used classifier.

The identification results for Hadiyya, Sidama, and Somali Wolayta in the Naïve Bayes unigram (n=1) feature set are 83.9%, 70%, 95.6%, and 74.8%, respectively compare with 1% mixture of Unigrams and Bigrams feature set for each language Hadiyya, Sidama, and Somali Wolayta is 93%, 92.7%, 99%, and 95.8% respectively. So these results showed that Naïve Bayes classifier with 1% mixture of Unigrams and Bigrams achieved highest accuracy and within the NB classifier the Somali language is the highest result for all other languages and has the greatest performance rate (99%).

In 60% mixture of Unigram & Bigram feature set for all classifiers with four targeted language classes, NB is an overall result of 91% and LR classifier 94%, RF, DT and GB classifier an overall result of 95.96%, 88.36% and 94.87% respectively. On this 60% mixture of Unigram & Bigram feature set of every utilized model showed an improvement in overall accuracy for that of unigram feature set. In the first experiment the random forest classifier exhibits the highest

performance among all classifiers and Somali Language is the highest predicted 98% it is high performance than other language.

In the second experiment, build a pipeline in a model and used n-gram range= (1, 3) mean It collects one word, two word and one, two, three words and analyzer='char' using character by character, Logistic regression has an overall performance of 98.9%, Random Forest classifier, with a performance of 98.5%, it is the second-best model overall. Thus, the outcome of logistic regression and random forest classifiers is nearly identical, but with logistic regression models exhibiting the highest performance at 98.9% Figure 5.13 shows screen shot of Logistic Regression classifier experiment.

In the third experiment, twenty feature sets were used as a column to display statistics about the data, with the test data used for training being varied in size. The average classification accuracy rate using Gaussian Naïve Bayes is 59.71%, whereas the rates for LR, RF, DT, and DT are 70.41%, 78.11%, and 76.69%, respectively.

Table 5.22 Somali language has the highest precision for our model NB classifier in Unigram & Bigram Mixture for top 1%.

In the first experiment's out of all used models the greatest level of accuracy result get in the RF Classifier Unigram & Bigram Mixture for the top 1% features in Somali language also 100%. Over all used Model the Logistic Regression Classifier with the best recall rate in Unigram & Bigram Mixture for top 1% features in Somali Language is 0.994%. Gradient Boosting and the Logistic Regression Classifier yield the second-highest precision among all the models in Unigram & Bigram Mixture for top 1% features in Somali Language also in both models it is around 0.989 % result.

According to the evaluation metrics mentioned above, the Logistic Regression Classifier with n-gram range of (1, 3) yields the maximum precision in Somali Language and Random Forest Classifier in Unigram & Bigram Mixture for top 1% features Somali Language is 100% precision result over all.

Finally, based on the results the hybrid approach leverages both Machine learning and Statistical approaches for better identification performance compared to using either approach individually. The third experiment individually approach twenty feature sets are low in accuracy for language classification but the second and the third experiments of used feature set are shows a good result it is suitable techniques for language classification.

## **6.2 Recommendation**

The study's findings are advantageous and applicable to all branches of linguistic identification and anybody interested in this field of study they can be used the findings and other resources. For this study we evaluated a model with various parameter values using our targeted language it is for Unigram (n=1), 1% mixture of Unigram & Bigram, 60% mixture of Unigram & Bigram and twenty feature sets used as a column. On this we find that it can be challenging to distinguish between two extremely similar languages like (Hadiyaa and Sidama), thus it would be best to apply new techniques for future study. In general according to the literature review, there hasn't been much or any research done on Ethiopian language but, it has been in a language with highly resources, such as European so further research in low resourced language.

## Reference

- [1] R. Řehůřek and M. Kolkus, “Language identification on the web: Extending the dictionary method,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5449 LNCS, pp. 357–368, 2009, doi: 10.1007/978-3-642-00382-0\_29.
- [2] T. Jauhiainen, M. Lui, M. Zampieri, T. Baldwin, and K. Lindén, “Automatic language identification in texts: A survey,” *J. Artif. Intell. Res.*, vol. 65, pp. 675–782, 2019, doi: 10.1613/JAIR.1.11675.
- [3] K. Ergetie, “Purpose Language Identification for Ethiopia Semitic Language Using Hybrid Approach,” no. November, 2018.
- [4] L. Grothe, E. W. De Luca, and A. Nürnberger, “A comparative study on language identification methods,” *Proc. 6th Int. Conf. Lang. Resour. Eval. Lr.* 2008, pp. 980–985, 2008.
- [5] M. Indhumathi, “Automatic Identification of Major Text Language,” *Turkish Online J. Qual. Inq.*, vol. 12, no. 7, pp. 7188–7199, 2021.
- [6] I. Models and I. Models, “Language Identification for very short texts : a review Introduction Datasets,” 2021.
- [7] L. W. Desta, “Modeling Text Language Identification for Ethiopian Cushitic Languages,” no. July, 2014.
- [8] M. Popović, “Complex Word Identification using character n-grams,” *Proc. 13th Work. Innov. Use NLP Build. Educ. Appl. BEA 2018 2018 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. NAACL-HTL 2018*, no. June, pp. 341–348, 2018, doi: 10.18653/v1/w18-0541.
- [9] L. Panich, “Comparison of Language Identification Techniques,” 2015, [Online]. Available: [https://dbs.cs.uni-duesseldorf.de/lehre/bmarbeit/barbeiten/ba\\_panich.pdf](https://dbs.cs.uni-duesseldorf.de/lehre/bmarbeit/barbeiten/ba_panich.pdf)
- [10] F. A. Smadja, “From N-grams to collocations an evaluation of xtract,” *Proc. Annu. Meet. Assoc. Comput. Linguist.*, vol. 1991-June, pp. 279–284, 1991.
- [11] M. Hasimu and W. Silamu, “On hierarchical text language-identification algorithms,” *Algorithms*, vol. 11, no. 4, pp. 1–27, 2018, doi: 10.3390/a11040039.
- [12] I. Weber, “Language Identification in a Highly Unbalanced Dataset by Signature : Date :,” no. December, 2022.
- [13] P. Mathur, A. Misra, and E. Budur, “LIDE: Language Identification from Text Documents,”

- 2017, [Online]. Available: <http://arxiv.org/abs/1701.03682>
- [14] R. Menon, “Detectsyt: A System for Detecting Language from the Text, Images, and Audio Files,” *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 10, no. 6, pp. 1975–1980, 2022, doi: 10.22214/ijraset.2022.44281.
- [15] B. Tadesse, “AUTOMATIC IDENTIFICATION OF MAJOR ETHIOPIAN LANGUAGES,” *Bdu.edu.et*, 2020, doi: <http://hdl.handle.net/123456789/10755>.
- [16] B. Duvenhage, “Short text language identification for under resourced languages,” *CEUR Workshop Proc.*, vol. 2540, no. NeurIPS, pp. 2015–2020, 2019.
- [17] A. Bhansali, A. Chandravadiya, B. Y. Panchal, M. H. Bohara, and A. Ganatra, “Language Identification Using Combination of Machine Learning Algorithms and Vectorization Techniques,” *2022 2nd Int. Conf. Adv. Comput. Innov. Technol. Eng. ICACITE 2022*, no. July, pp. 1329–1334, 2022, doi: 10.1109/ICACITE53722.2022.9823628.
- [18] A. Avenberg, “Automatic language identification of short texts,” no. September, 2004, [Online]. Available: <http://www.teknat.uu.se/student>
- [19] V. Križ, M. Holub, and P. Pecina, “Feature extraction for native language identification using language modeling,” *Int. Conf. Recent Adv. Nat. Lang. Process. RANLP*, vol. 2015-January, pp. 298–306, 2015.
- [20] A. Kulmizev et al., “The power of character n-grams in native language identification,” *EMNLP 2017 - 12th Work. Innov. Use NLP Build. Educ. Appl. BEA 2017 - Proc. Work.*, pp. 382–389, 2017, doi: 10.18653/v1/w17-5043.
- [21] N. B. Sristy, B. S. Krishna, N. S. Krishna, and V. Ravi, “Language identification in mixed script,” *ACM Int. Conf. Proceeding Ser.*, pp. 14–20, 2017, doi: 10.1145/3158354.3158357.
- [22] A. H. Kulkarni, P. S. Upparamani, R. J. Kadkol, and P. V Tergundi, “Script Identification from Multilingual Text Documents,” vol. 4, no. 6, pp. 15–19, 2015, doi: 10.17148/IJARCCE.2015.4605.
- [23] A. F. Hidayatullah, R. A. Apong, D. T. C. Lai, and A. Qazi, “Corpus creation and language identification for code-mixed Indonesian-Javanese-English Tweets,” *PeerJ Comput. Sci.*, vol. 9, pp. 1–24, 2023, doi: 10.7717/PEERJ-CS.1312.
- [24] K. Indhuja, M. Indu, C. Sreejith, and P. C. R. Raj, “Text Based Language Identification System for Indian Languages Following Devanagiri Script,” vol. 3, no. 4, pp. 327–331, 2014.
- [25] A. L. Tonja et al., “Natural Language Processing in Ethiopian Languages: Current State, Challenges, and Opportunities,” *4th Work. Resour. African Indig. Lang. RAIL 2023 - Proc. Work.*, pp. 126–139, 2023, doi: 10.18653/v1/2023.rail-1.14.

- [26] S. Nisioi, “Feature analysis for native language identification,” *Lect. Notes Comput. Sci.* (including *Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics*), vol. 9041, pp. 644–657, 2015, doi: 10.1007/978-3-319-18111-0\_49.
- [27] C. Bartz, T. Herold, H. Yang, and C. Meinel, “Language identification using deep convolutional recurrent neural networks,” *Lect. Notes Comput. Sci.* (including *Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics*), vol. 10639 LNCS, no. 1, pp. 880–889, 2017, doi: 10.1007/978-3-319-70136-3\_93.
- [28] M. Lui and T. Baldwin, “langid.py: An off-the-shelf language identification tool,” *Proc. Annu. Meet. Assoc. Comput. Linguist.*, no. July, pp. 25–30, 2012.
- [29] F. Gaim, W. Yang, and J. C. Park, “GeezSwitch: Language Identification in Typologically Related Low-resourced East African Languages,” *2022 Lang. Resour. Eval. Conf. Lr. 2022*, no. June, pp. 6578–6584, 2022.
- [30] O. Giwa and M. H. Davel, “Text-based language identification of multilingual names,” *Proc. 2015 Pattern Recognit. Assoc. South Africa Robot. Mechatronics Int. Conf. PRASA-RobMech 2015*, pp. 166–171, 2015, doi: 10.1109/RoboMech.2015.7359517.
- [31] G. R. Botha and E. Barnard, “Factors that affect the accuracy of text-based language identification,” *Comput. Speech Lang.*, vol. 26, no. 5, pp. 307–320, 2012, doi: 10.1016/j.csl.2012.01.004.
- [32] E. Chávez, M. García, and J. Favela, “Fast and Accurate Language Detection in Short Texts using Contextual Entropy,” *Res. Comput. Sci.*, vol. 90, no. 1, pp. 351–358, 2015, doi: 10.13053/rcs-90-1-27.
- [33] A. F. Hidayatullah, A. Qazi, D. T. C. Lai, and R. A. Apong, “A Systematic Review on Language Identification of Code-Mixed Text: Techniques, Data Availability, Challenges, and Framework Development,” *IEEE Access*, vol. 10, no. November, pp. 122812–122831, 2022, doi: 10.1109/ACCESS.2022.3223703.
- [34] V. Ramanarayanan, R. Pugh, Y. Qian, and D. Suendermann-Oeft, “Automatic turn-level language identification for code-switched Spanish–english dialog,” *Lect. Notes Electr. Eng.*, vol. 579, pp. 51–61, 2019, doi: 10.1007/978-981-13-9443-0\_5.

## Appendix

### Annex 1- Python Code: Predicting with Real Data (Out of Vocabulary Word)

```
defdetect_language(Text):  
  
#vectorize the text  
  
test = top1PrecentMixtureVectorizer.transform([Text])  
  
var_test=toNumpyArray(test)  
  
rf= clf8.predict(var_test)  
  
#Check for the prediction probability  
  
pred_proba=clf8.predict_proba(var_test)  
  
pred_percentage_for_all=dict(zip(clf8.classes_,pred_proba[0]))  
  
print("Prediction using Random Forest Top 1%: : {}, Prediction Score :  
{}".format(rf[0],np.max(pred_proba)))  
  
print()  
  
print(pred_percentage_for_all)
```

### Out put

```
#test text in wolayta  
detect_language('gallassaataaniamottabeikke. Ta doonaappe')  
Prediction using Random Forest Top 1%: : Wolayta , Prediction Score : 1.0  
  
{'Hadiyya': 9.658346166325205e-57, 'Sidama': 2.3837862713507258e-52,  
'Somali': 8.472191788092398e-42, 'Wolayta': 1.0}  
#test text in Somali  
detect_language('BilowgiiIlaahsamadaiyodhulkuuabuuray. Dhulkunaqaab ma lahayn, wuuna')  
Prediction using Random Forest Top 1%: : Somali , Prediction Score : 1.0  
  
{'Hadiyya': 5.420584570147611e-121, 'Sidama': 8.360830686193343e-91,  
'Somali': 1.0, 'Wolayta': 4.432196915649767e-87}
```

## Annex 2-Python Code Classification Error Analysis

```
defplotTopErrors(y_predict, top=5):

ys = y_test.values

Xs = x_test.values

errorCount = 0

fori in range(len(ys)):

if not ys[i]==y_predict[i]:

errorCount += 1

print("#{}: Expected: {}, Predicted: {}".format(errorCount, ys[i], y_predict[i]))

print("Text:", Xs[i])

print("=====")

iferrorCount>= top:

break
```

### Output of Error Analysis

```
#1: Expected: Sidama, Predicted: Hadiyya
Text: HagiiruSokka
=====
#2: Expected: Hadiyya, Predicted: Sidama
Text: YihuxxiadilhannonnetteBoollaankaelliinchifiranne
=====
#3: Expected: Sidama, Predicted: Hadiyya
Text: seekkitinebuuxxe
=====
#4: Expected: Wolayta, Predicted: Sidama
Text: Baabilooneeshuchchadoorenneworakanati
=====
#5: Expected: Wolayta, Predicted: Hadiyya
Text: Ermmaasa
```

---



### Annex3-Screenshot Python Code Most Common Chars in Unigram

```
In [18]: # get most common chars for a few ethiopian languages
ethiopianLanguages = ['Hadiyya', 'Somali', 'Sidama', 'Wolayta']
relevantChars_OnePercent = getRelevantCharsPerLanguage(unigramFeatures, language_dict_unigram, 1e-2)

# collect and sort chars
ethiopianCharacters = []
for lang in ethiopianLanguages:
    ethiopianCharacters += relevantChars_OnePercent[lang]
    ethiopianCharacters = list(set(ethiopianCharacters))
    ethiopianCharacters.sort()

# build data
indices = [unigramFeatures.index(f) for f in ethiopianCharacters]
data = []
for lang in ethiopianLanguages:
    data.append(language_dict_unigram[lang][indices])

#build dataframe
df = pd.DataFrame(np.array(data).T, columns=ethiopianLanguages, index=ethiopianCharacters)
df.index.name = 'Characters'
df.columns.name = 'Languages'
```

#### Annex 4-Screenshot Python Code Most Common Chars in Bigram

```
n [167]: # get most common chars for a few ethiopian languages
ethiopianLanguages = ['Hadiyya', 'Somali', 'Sidama', 'Wolayta']
relevantChars_OnePercent = getRelevantCharsPerLanguage(bigramFeatures, language_dict_bigram, 1e-2)

# collect and sort chars
ethiopianCharacters = []
for lang in ethiopianLanguages:
    ethiopianCharacters += relevantChars_OnePercent[lang]
    ethiopianCharacters = list(set(ethiopianCharacters))
    ethiopianCharacters.sort()

# build data
indices = [bigramFeatures.index(f) for f in ethiopianCharacters]
data = []
for lang in ethiopianLanguages:
    data.append(language_dict_bigram[lang][indices])

#build dataframe
df = pd.DataFrame(np.array(data).T, columns=ethiopianLanguages, index=ethiopianCharacters)
df.index.name = 'Characters'
df.columns.name = 'Languages'
```