ADDIS ABABA UNIVERSITY

COLLEGE OF NATURAL AND COMPUTATIONAL SCIENCES

SCHOOL OF INFORMATION SCIENCE

A COMPARATIVE STUDY OF AUTOMATIC LANGUAGE IDENTIFICATION OF ETHIO-SEMITIC LANGUAGES

REDIAT BEKELE ASFAW (MR.)

June 2018

Addis Ababa, Ethiopia

ADDIS ABABA UNIVERSITY
COLLEGE OF NATURAL AND COMPUTATIONAL SCIENCES
SCHOOL OF INFORMATION SCIENCE

A COMPARATIVE STUDY OF AUTOMATIC LANGUAGE IDENTIFICATION OF ETHIO-SEMITIC LANGUAGES

BY:

REDIAT BEKELE ASFAW (MR.)

A Thesis submitted in partial fulfilment of the requirements for the Degree of Master of Science in Information Science

June 2018
Addis Ababa, Ethiopia

ACKNOWLEDGMENTS

I would like to thank the God and Father of Jesus Christ for helping me settle my thoughts and create a space within me to pick up the research paper I parked over 4 years ago.

I can't thank enough Ms. Samrawit Girma Beyene, who has been a help continually reminding me to complete my research thesis and support me psychologically, financially and colour printing this research paper.

I cannot pass without acknowledging the personal follow up and encouragements of Mr. Mihretu Tekalign and Mr. Solomon Lemlem that kept me going to finish my work on this research paper.

I am also indebted to Solomon Teferra Abate (PhD) for his supportive assistance to continue this research and all the way towards its completion.

I am glad to have Wondwossen Mulugeta (PhD) as my examiner, as he provided an insightful advice during the research proposal defence that helped me follow the right direction on this research.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACRONYMS AND ABBREVIATIONS

AAU	Addis Ababa University
ANN	Artificial Neural Network
BL	Language model generated from fixed length character n
BOW	Bag of Words
BY	Language model generated N-grams generated from text string of training and test corpus, Byte order
CBN	Character Based N-gram
CFA	Cumulative Frequency Addition
CLIR	Cross Language Information Retrieval
F	F-score
FL	Language model generated from Fixed length character n-gram from BOW with location features
FN	False Negative
FP	False Positive
ICT	Information Communication Technology
IL	Language model generated from Infinity n-gram on BOW with location features
IN	Language model generated from Infinity n-gram on BOW without location features
LID	(Automatic) Language Identification
NBC	Naive Bayes Classifier
NLP	Natural Language Processing
NTLK	Natural Language Toolkit
OCR	Optical Character Recognition
P	Precision
R	Recall
TN	True Negative
TP	True Positive
VSM	Vector Space Model

ABSTRACT

The dominant languages under the family of Ethio-Semitic languages are Amharic, Geez, Guragigna and Tigrigna. From the findings of the language identification studies on European languages, there is a conclusion that most classifiers performance reached the accuracy of 100%. Local and global studied confirmed that Naïve Bayes Classifier (NBC) classifier does not reached the accuracy level of 100% in language identification especially on shorter test strings. Comparative Language Identification studies in European languages shows that Cumulative Frequency Addition (CFA) performs close to 100% accuracies better than the NBC classifier.

The purpose of our study is to assess the performance of CFA as compared to NBC on Ethio-Semitic languages, to validate the research findings of CFA and NBC classifiers, and recommend the classifier, language model, evaluation context and the optimal values of N that performs better in language identification.

In this research we have employed and experimental study to measure the performance CFA and NBC classifiers. We have developed a training and test corpus from online bibles written in Amharic, Geez, Guragigna and Tigrigna to generate 5 different character based n-gram language models. We have measured the classifiers performance using under two different evaluation contexts using 10-fold cross validation. F-score is used as an optimal measure of performance for comparing classifiers performances.

The classifiers commonly exhibited higher performance when the length of the test phrase grows from a single word to 2, 3 and beyond to reach an F-score measure beyond 99%. Both classifiers performed similarly under each context corresponding to the language models and n-grams tested. The language model, fixed length character n-grams with location features, exhibited highest performance in F-score for both classifiers under each evaluation contexts on test strings as short as one word length. N=5 on Fixed length character n-grams with location features language model is the optimal value of N whereas N=2 is the optimal value for the remaining language models on both CFA and NBC classifiers and evaluation contexts. Based on our findings CFA is a classifier that performs better as compared to NBC as it is founded in sound theoretical assumptions and its performance in language identification.

Keywords: language identification, Naïve Bayes, N-gram, Cumulative Frequency Addition, Fixed length character N-grams, Infiniti n-grams, N-gram from text string

CHAPTER ONE

INTRODUCTION

1.1 Background

Today, the need for multi-language communication applications to serve people from different nations in their native languages has gained an increasing importance and a very wide application. Towards this end, software applications and cloud services are becoming language agnostic (i.e. work with many languages) to serve a wider community accessing them from diverse language and cultural background.

Search engines powered with Automatic Language detection mechanisms enabled to provide options for translation to a selection of a second language by users, social media sites such as Facebook, are providing options to translate a content from posts and comment boxes to the default language settings of the user account. Web browser applications automatically detecting the language of a web page and translates it on the fly to the default language selection of the user. Behind all these, Natural Language Processing (NLP) in general and Automatic Language Identification (LID) in particular plays a significant role at the pre-processing stage of these multi-language systems and applications.

Taking the story home, in Ethiopia, we are still at an infant stage in developing a retrieval system for documents written using the Ethiopic writing system, locally known as Fidel. At the time of writing this paper, the application of language detection and translation for text written in Amharic language on major world search engines such as Google is still at an infant stage. Major social media sites like Facebook are not yet providing an option for neither of Amharic, Tigrigna or Guragigna as a user language. Currently, considering a Cross Language Information Retrieval (CLIR) and development of multilingual systems between Amharic, Geez, Guragigna and Tigrigna is under study in the academic sphere. We believe the next stage of development will lead to creating a retrieval system that can take a multilingual query and able to retrieve documents from the corresponding corpus that the system detect the language in advance. Towards this, Automatic Language Identification adds capabilities to Search

Engines, Social Media sites and web browser applications to detect content written in either of Amharic, Tigrigna or Guragigna languages and offer proper translation options.

Automatic Language Identification is the process of utilizing automated tools to identify the natural language a digital document is written. It is also called Language Detection or Language Recognition. A general paradigm in automated language identification is to create language models during a training phase and compare input documents against these models during language identification also called classification phase. This places the task into the domain of supervised learning methods. Another consequence is that the set of target languages needs to be known beforehand, which makes language identification a classification problem [4].

Correct identification of the language both in query and documents is a prime importance for search engines that works with multi-lingual corpus. Not only for search engines but also applicable to all systems that depend on language detection such as Automatic language translation systems. Therefore, Language identification helps to assist Cross Language Information Retrieval (CLIR) systems to become language agnostic and function well in finding relevant documents working with both multilingual queries and corpus [1].

There are two kinds of approaches for Automatic Language Identification. The first is the linguistic approach, which requires prior linguistic knowledge and involves the construction of linguistic resources, such as the dictionary method, which constructs language models based on word relevance.

The second approach is using the probabilistic and statistical approaches, which uses the knowledge built automatically from the text corpus representative of the language. The goal of this approach is to use statistical models and probabilities to capture certain regularities of the languages and their associated frequency or probability of occurrences. These empirical regularities captured using the frequency or probability of occurrences can be used as the linguistic knowledge. The language identification process is to calculate the probability of a given text to belong to either of a given set of languages, based on the regularities observed. The biggest challenge in dealing with language identification problems is the lack of standard or a publicly available data set, corpus. Creating such data set is highly expensive, as it requires

an extensive work using human annotators [2]. As a result most language identification researches are conducted based on test and training corpus developed for the intended research purpose. Similarly we have followed a similar path towards gathering relevant data for training and testing purpose for this research.

Different tools are developed to classify text to different languages based on approaches that work in a supervised manner. I.e. for a given sample of each language, similarity measures are used as a basis for prediction and the text is classified according to its similarity with the learning text sets.

Using a supervised model has its own limitations, as it cannot identify a language that is not contained in the training set. However, in situations where all the languages of documents in the collection are known in advance and a language model is developed for each language, the supervised method is applied [2].

Statistical models are generic models and, using machine learning, they utilize features from training samples to categorize text in to the languages known in advance. Several methods of feature extraction have been used for language classification, including unique letter combinations, short word method, N-gram method, and ASCII vector of character sequences. Among the most reported classifiers are Bayesian Decision Rules, Rank Order Statistics, K-Nearest Neighbour, and Vector Space Model.

Naïve Bayes Classifier is a widely researched and applied classification method in most classification problems for its simplicity of implementation and classification performance. It is also studied locally for language identification on Ethio-Semitic languages [6].

Researches propose for a new classification technique, which is simpler than the conventional Naive Bayesian classification method, performing similarly in speed but better in accuracy to detect a language of digitized documents. This proposed technique is called Cumulative Frequency Addition (CFA) which we study in this research [7]. As compared to the Naïve Bayes Classifier (NBC), CFA is relatively new classification method with little research on the

area of language identification. Our study describes a comparative study of this classifier against the Naïve Bayes Classifier when applied on Ethic-Semitic languages [7].

We have tried to reach Kidist [6] to use the training and testing corpus she developed for her research work. As we could not reach her, we have developed a different corpus for training and test purposes of the classifiers we developed for this research.

We have selected Amharic, Geez, Guragigna and Tigrigna for this research as these languages represent the larger language speaking groups from Ethio-Semitic language both in urban and rural communities to conduct their day-to-day activities. In addition to this, there is also available digital text document for use for training and testing purposes to conduct research.

As compared to the Amharic, Guragigna and Tigrigna, currently Geez is a dormant language used within the limited group of people mostly in the religious circles, with a limited set of growing audience and practical usage to conduct the day-to-day activities of the speakers. Considering the huge historical resources written in this language and the growing interest to study the language as a communication medium, we have included it as an important piece in our research.

In this research, we are interested to see the performance of the Cumulative Frequency Addition (CFA) classification method in comparison with the Naïve Bayes Classifier (NBC) based on character based n-gram language modelling in automatically detecting documents written in Amharic, Geez, Guragigna and Tigrigna.

We believe that depending on the availability of written digitized text to use, this research can be extended to include other Ethio-Semitic languages that use the Ethiopic characters as their system of writing.

1.2 Statement of the problem

Language profile building using n-grams for the purpose of Information Retrieval is not a new concept. It has been applied to European and Asian languages for long to develop Language Identification and Retrieval Systems. According to McNamee [16], Language identification is

a solved problem, because most classification methods tested, achieved accuracies approaching 100% on tests comprised of European languages. Coming home, the application of n-gram as feature type for language and document profiling for the purpose of building a language identification system for Ethio-Semitic language has limited research.

Studying the latest research conducted at Jimma Institute of Technology, which is the only local research we can find on language identification for Ethio-Semitic languages, Kidist [6] used the Hybrid of Machine Learning and Rule Based approach using Bayesian Classification Method and achieved the maximum average F-score of 88.75% on unspecified length of test string. Similar researches conducted on European languages exhibited 88.66% for shorter test strings (50 characters long) based on Naïve Bayes Classification [7]. The performance of Cumulative Frequency Addition technique on European languages showed promising results by improving performance beyond 97.59% for shorter strings on the same training and test data that Naïve Bayes Classifier achieved 88.66% accuracy. Based on this findings in the European languages, we hypothesize that Cumulative Frequency Addition performs better for language identification on Ethio-Semitic languages.

According to Bashir Ahmed, Sung-Hyuk Cha, and Charles Tappert [7], using Cumulative Frequency Addition of N-grams we can achieve better performance over Naive Bayes Classifier when applied to fixed length n-grams without normalization of the training data. We hypothesize that by using Cumulative Frequency Addition, we can skip normalization of homophones on the training set as non-contributing factor to improve the performance of the classification model as exhibited on the study made on European languages [7].

This research is set out to answer the following questions:

1. Does CFA lead to better performance as compared to NBC on Ethio-Semitic languages?
2. What is the effect of different Character Based N-grams language models on the performance of CFA and NBC classifiers?
3. What is the effect of different evaluation contexts on the performance of CFA and NBC classifiers?

1.3 Objective(s)

1.3.1 General Objective

The general objective of this study is to explore the application of Cumulative Frequency Addition (CFA) in automatically detecting the language of documents written in Amharic, Geez, Guragigna and Tigrigna languages and asses the factors that affect the performance of this classification method in comparison to the Naïve Bayes Classifier (NBC).

1.3.2 Specific Objectives

The specific objectives identified to achieve the general objective are as follows:

1. Develop a training and testing data sets for Amharic, Geez, Guragigna and Tigrigna for modelling and testing purposes.
2. Develop different language models using fixed length character n-grams applied on Bag of Words (BOW) and on Text without tokenization, and infinity character n-grams.
3. Implement CFA and NBC classifiers following the theoretical conceptions
4. Conduct performance measurements using Precision, Recall and F-score to evaluate the classifiers in detecting languages
5. Identify the language model, the optimal values of N and evaluation context that provides for highest performance for CFA and NBC classifiers on Ethio-Semitic languages
6. Recommend a better performing classifier, language model, evaluation context and the optimal value of N.

1.4 Scope and Limitations

The scope of this project is limited to digital documents written in Amharic, Geez, Guragigna and Tigrigna languages selected from Ethio-Semitic languages. The remaining Ethio-Semitic languages such as Argobba, Harari, Dahalik and etc. are not included in our research for limited access to digitized documents at the time of data collection. As there is no standard corpus to use, this research used texts extracted from the Bible written in Amharic, Geez, Guragigna and Tigrigna for training and testing.

Due to the resource constraints, the time and storage complexity of classifiers, language models and evaluation contexts not analysed on this research. This research does not consider cases when a text written partly in one language and partly in another language where the dominant language of the input document is automatically detected.

The resource constraints also limited the number of classifiers we have considered in our study. We considered CFA applied to fixed length and infinity character n-grams language profiling for language classification and compare the results with the NBC findings on the same training and test data. Due to the Memory and processing power limitation of our development and testing machine a reasonable corpus size is used for training and testing purpose.

1.5 Significance of the Research

This research is beneficial in terms of testing a language identification model based on CFA. It will provide the assessment of CFA classifier on language detection of the less researched Ethio-Semitic languages, i.e. Amharic, Geez, Guragigna and Tigrigna. In this document CFA tested over different language models and its performance is compared with the most researched and widely used Naive Bayes classifier. The outcomes of this research will add to the existing knowledge of language identification, Information Retrieval and other Text Classification task on Ethio-Semitic languages and give insights and directions to further researches that will be conducted around Language Identifications on Ethio-Semitic languages.

1.6 Methodology

In this research we have followed experimental study and performed measurements of classifiers performance under different language models and evaluation context.

The following methods are employed in order to achieve the objectives of this study.

1.6.1 Literature review

Related researches in the automatic language identification, information retrieval and document classifications are consulted as an input for this research.

1.6.2 Data Collection

For each languages, we have extracted 300Kb of text from online bibles written in Amharic, Geez, Guragigna and Tigrigna.[1] We have performed text processing to remove punctuation marks and non-Ethiopic alpha numeric characters. After the data cleaning the corpus consists of an average of 100Kb text for each language.

1.6.3 Model and Classifiers Development

Due to the different character encoding and unique features of the languages we studied, employing the available NLP tools such as Natural Language Toolkit (NTLK) usually used for NLP problems in many European languages, have limitations to directly use for Ethio-Semitic languages. Therefore, we have developed a project code to perform our experimental study.

Two classifiers, CFA and NBC are developed using their theoretical conceptions. Five different language models are developed based on Character Based N-gram (CBN) language modelling approach.

There are different language modelling approaches. These are following either or both of the word based and character based approaches. Classifiers are build using the word based approach to perform text classification problems where relatively large volumes of text is used for classification and language identification. When it comes to language identification considering very short texts as inputs, powerful language identifiers can be build using character based n-grams used for language modelling. With character based n-grams, many features can be extracted from a training and test document as compared to the word based approach.

An n-gram character is a sequence of n consecutive characters in a document. An n gram for any document is obtained by moving a window of n boxes in the text. This movement is made

[1] Documents written in Amharic language are from the 2007 edition of the Amharic bible taken from http://listen.bible.is/AMHSDV/Rom/1. The Tigrigna content are from the 1991 edition of the Tigrigna bible available on http://listen.bible.is/TIRUBS/Rom/1. Contents for Guragigna are from the 1982 edition of the bible available on http://listen.bible.is/SGWBSE/Rom/1. Documents written in Geez are from https://www.stepbible.org/version.jsp?version=Geez. We have assumed that the version differences of the bible documents only indicate the year of publication of the documents but not necessarily mean a version difference in the languages themselves.

in stages; for the case of a character, one stage corresponds to one character, and a word for n-grams of words. We count the frequencies of n-grams found from this exercise to build language models [3]. In this research, n-grams of characters is used, thus an n-gram refers to string of n consecutive characters. The window of n runs from 2 to 5 on fixed length character n-grams.

Following the observation that each language has some characteristic n-grams which appear frequently, we compare the frequency of n-grams in a previously unseen text with those of training texts for different language classes.

The Microsoft Visual Studio Code with Python 3 used to develop language models, implement the classification algorithms, and performance testing. Input files such as training and test corpus and output files, such as the language models and test results are saved as UTF-8 encoded text documents.

We have used Compaq Presario CQ40 laptop with Core2Duo processor, 2GB RAM and 160GB HDD running on Ubuntu 16.04LTS operating system as a development machine. The development environment is Visual Studio Code v1.23.1 installed as part of the 64 bit Anaconda 3, with Python v3.6.4. A Python package, MatPlotLib v2.2.2, is used for graphical presentation of performance measurements.

1.6.4 Experiments

This research intends to utilize the CFA and NBC classifiers on language models developed based on Character Based N-gram (CBN) features. It evaluates the performance of the classifiers in two different evaluation contexts, using all the available n-gram features within a range of n=2, 3, 4 and 5 for fixed length n-gram models and using all the available n-gram features within a range of n=2, 3, 4 … length (word) for infinity n-grams as Context I, and independent evaluation of each n=2, 3, 4 and 5 as Context II.

90.44% of the total corpus used for training and testing consisted of words with the maximum length of 5 characters. For this reason, we have selected the values of N to run up to N=5 for language models based on fixed length character N-grams.

The CFA and NBC classifiers need to be trained on training documents for each language, Amharic, Geez, Guragigna and Tigrigna. For the evaluation purpose, considering the language models as factors affecting performance, the following language models are selected for our experiment.

1. Exp-1 – Language model generated from fixed length character n-grams without location features extracted from bag of words collected from language training and testing corpus, which we refer to it as Baseline Model and abbreviated as BL.

2. Exp-2 - N-grams generated from text string of training and testing corpus, which we refer to it as N-gram from Text or Byte order Model and abbreviated as BY

3. Exp-3 – Fixed Length N-grams generated from bag of words with location features, which we refer to it as Fixed Length with Location Feature Model and abbreviated as FL

4. Exp-4 - N-grams generated from Infinity n-grams of bag of words without location features, refer to it as Infinity n-gram without Location Feature Model and abbreviated as IN

5. Exp-5 - N-grams generated from Infinity n-grams of bag of words with location features, refer to it as Infinity n-gram with Location Feature Model and abbreviated as IL

The performance of CFA on each language model is evaluated against the NBC classifier and the better performing approach in terms of Precision, Recall and F-measure will be identified.

1.7 Evaluation

The outcome of the study is evaluated the performance of classifiers across different language models and evaluation contexts. Each classifier is tested using unseen documents withheld from the training corpus. The predicted language is checked against the language category of each testing document and the number of times the classifier predicts the language is tabulated using Confusion Metrix to calculate the performance metrics, Precision, Recall and F-score language models. The best performing classifier is identified using F-score for comparative measurements of classifiers performance.

1.8 Thesis Outline

This researched is organized in five chapters. In Chapter 2 we have reviewed relevant researches related to Automatic Language Identification which is a Text Classification problem. In Chapter 3 we discussed on the design and implementation of our classifier based on CFA and NBC. Chapter 4 discusses the experimentations we have done and analyses the results of tests. Chapter 5 details the conclusions, recommendation and forwards for future researches.

For concise presentation of this research document, easy accessibly and independent verification, the training and test corpus and the language models and project code is availed on address https://github.com/Rediat/cfa.git. Excerpts of corpus, project code, language model, testing strings and classification module outputs are attached as appendix with appropriate locations in the project folder shared online.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

We are living in a world where people speak their native languages to communicate among themselves and others. Advances in ICT brought applications and services to aid the production and communication of information easily across geographical and language boundaries. The wide adoption of Social Medias also enabled users to create business, casual and intimate relations among people located in opposite sides of the globe having different cultural and language background. Communications with text, audio and video is also become a common place between people using real time language identification and translation systems, such as Skype.

Living in this technologically advanced world, the local languages used in Ethiopia have not taken the opportunities that comes with these applications and services to allow speakers of the language to take advantage of the technological advances in Natural Language Processing (NLP) for there are limited researches around Ethio-Semitic and other local languages in Ethiopia.

Automatic language identification is based on the understanding that, words and character used in a language are not fully random but according to Zipf law [8] follow certain regularities. Since this law apply to all languages, using the frequencies of the words and character distribution to model languages and with the help of available statistical classifiers, we can do automatic language identification on documents written in Ethio-Semitic languages.

2.2 What is language identification?

Language Identification is the process of finding in which natural language the contents of a digitized text is written. Language identification is an extensive research area in Natural Language Processing domain such as machine translation, information retrieval, summarization and question answering etc. which require prior language identification before processing. Language identification falls into two approaches: one is non-computational; and the other is computational [9].

Non-computational approaches requires the researchers to have sufficient knowledge about the language to be identified such as diacritics and symbols, most frequent words used, character

combinations etc. while computational approaches rely on statistical techniques instead of linguistic knowledge to solve language related problems using large training dataset.

Statistical approaches follow two successive phases: one is the Training phase and the other is Classification phase. In the training phase, feature extraction is performed from the given training dataset, known as the training corpus.

After generating the models, the input document model is compared to the models generated from the training set at the classification stage and the distance between them is measured with the help of classification techniques. The language model that has the minimum distance to the input document represents the language of the document [11].

Written text consists of words, spaces and punctuation marks. The building blocks of words are characters following the writing system of a language. To model or develop and language profile for a specific language document, we have the length or frequencies of words and character n-grams to extract and use from each language. Length can be fixed at n number of characters depending on the researcher's interest [9].

2.3 Language Identification Models

Language identification models contain entities (words or n-grams) for extracting language specific features. Depending on the frequency of occurrence, these features are listed as entities in a language or document model [9]. In the following, we present the most widely used approaches used as language modelling techniques.

2.3.1 Unique letter combinations

This method of feature extraction is the use of unique letter combinations characteristic of a language to use for language modelling. This method works by enumerating a number of short sequences which are unique to a particular language.

Table 2.1 indicates an example of the list of characteristic sequences proposed by various researchers as unique to European languages.

Language String	Unique String
Dutch	vnd
English	ery
French	eux
Gaelic	mh
German	der
Italian	cchi
Portuguese	seu
Serbo-croat	lj
Spanish	ir

Table 2.1 Table depicting a sample of unique strings in European languages

Clearly these particular strings do not serve as reliable differentiators as no one could claim that a text that discussed zucchini or Pinocchio would have to be Italian. In addition to that, any borrowing of geographical terms such as Montreux or the simple adoption of words such as milieux into English would make this tests very weak and ineffective to build powerful language identification system based on this model [12]. The problems lie in the fact that such a unique strings approach depends on the presence of few strings, and that it fails to weight the evidence that it finds. This problem necessitates considering all observed strings using n-gram methods and an optimal weighting scheme used in classifiers such as Bayesian decision rule system.

2.3.2 Common Words or Frequent Words Approach

This is a type of word-based approach that generate the language model using a specific amount of the most frequent words [9]. These words describe a set of words having the highest frequency of all words occurring in a text to generate a language model. Since such common words make up substantial amounts of the text, this approach works relatively well as compared to unique letter combinations approach, if enough text is available to be classified. When the text to be classified is very short, there is a substantial chance that it will not contain any of the common words making classifiers less powerful to detect a language [12]. Even on longer text strings containing much of rare words and proper names a classifier based on common word approach is not expected to work well.

Additionally, since common words tend to be short, systems such as the n-gram which statistically weight the evidence provided by the short strings found in text will naturally

incorporate what information is available from short words. They will also be able to incorporate the information provided by common suffixes and prefixes. Furthermore, the common word methods are somewhat limited in that they depend on the ability to define and recognize common words. If tokenization into words is difficult (as in Chinese), or if defining a set of common words is difficult then the common word approach may be difficult or impossible to apply [12].

2.3.3 Short Word-Based Approach

The short word-based approach is similar to the common words/frequent words method, but it only uses words up to a specific length to construct the language model, independent from the particular word frequency [8].

This is based on tokenizing and extracting all words with a length of up to some number of characters for each languages under study that occurred at least for given number of times taken as threshold. The idea behind this approach is the language specific significance of common words like conjunctions and function words having mostly a shorter lengths [9]. Classifiers based on this models tend to work better for larger documents but when the text to be classified is very short, this make them less powerful to detect a language [12].

2.3.4 N-Gram-Based Approach

An n-gram is a contiguous n-characters generated from string of text or a word. The beginning and the end of a word are often marked with an underscore or a space before N-grams are created. This helps to discover start and end N-grams at the beginning and ending of a word and to make the distinction between them and inner-word N-grams. For instance, the word data, surrounded with the underscores, results in the following [6]:

N-grams:

Unigrams: _, d, a, t

Bigrams: _d, da, at, ta, a_

Trigrams: _da, dat, ata, ta_

Quad grams: _dat, data, ata_

5-grams: _data, data_

6-grams: _data_

To detect the language of a document, at first its N-gram language model is created. Commonly, pre-processing is employed, i.e. punctuation marks are deleted. Moreover, they are tokenized and surrounded with spaces or underscores. From these tokens, N-grams are generated and their occurrences are counted. The list of N-grams is sorted in descending order of their frequencies and the most frequent ones produce the N-gram language model of the document.

The main advantage of the N-gram-based approach for language modelling is in splitting all text strings and words in to slices shorter than the number of characters in a word. Errors coming from incorrect user input or Optical Character Recognition (OCR) failures remain only in some of the N-grams while leaving other N-grams generated for the same word/text unaffected. This improves correctness of the language models generated for the languages. However, N-grams of small length such as N=1 or N=2 are not very distinctive and some of them are present in language models of many languages, particularly for those very closely related languages.

To handle this problem, we have also considered N-grams of 3, 4 and 5. In addition to this, we have also studied a less researched approach which enable to extract large number of character N-gram features for a particular word called All-strings or Infinity N-gram approach with minor modifications of using the N-grams starting from N=2. That is, a bag of N-grams in which N=2 up to the length of a word in characters count.

2.4 Language Classification Methods

Classification is the second step in the language identification process. The language of a document is identified using the language models for the classification methods. Different statistical classification approaches can be used for language identification as discussed below.

2.4.1 Vector Space Model

Prager [17] applied the Vector Space Model (VSM) based on the similarity computation via the cosine distance between the training and test language model. The numerical values within one vector are defined by a token's occurrence in the training set times its inverse document frequency. Prager [17] used three different values for the idf-weight: 1/ni, $1/\sqrt{ni}$ and $1/\log(1 + ni)$. The best performing of these was the first, 1/ni [9].

2.4.2 Markov Chains in combination with Bayesian Decision Rules

The basic idea behind Dunning [12] approach is the computation of a token's occurrence in every system supported language. The language of every language model, constructed using Markov Chains, is computed using the Bayesian Decision Rule. All supported languages are possible events given one considered test language model. The most likely language for a given language model is computed by the probability for one language L_i within the language pool L times the probability of the language model given language L_i computed with Markov Chains.

2.4.3 Monte Carlo sampling

Poutsma [18] involved Monte Carlo sampling on language identification to compare it to the Ad-Hoc Ranking from Cavnar and Trenkle [19] and the Mutual Information Statistics from Sibun and Reynar [20]. Instead of the necessary generation of language models in order to classify the documents language, the Monte Carlo technique used by Poutsma utilizes dynamic models. These are built by randomly selected features [9]. The selection iteration is executed until the amount of features is adequate enough to determine the entire documents language. The necessary amount of features is investigated calculating the standard error σ of the feature samples.

2.4.4 Relative Entropy

Sibun and Reynar [20] applied Relative Entropy also called Kullback-Leibler distance. The language models describe probability distributions and the Relative Entropy computes their similarity based on the amount necessary encoding information for a second language model given a first one. The respective language model probability distributions are computed by determining the amount of a group of token like trigrams, for instance. The language a language model might be written in most likely will minimize the Relative Entropy the language's training model.

2.4.5 Ad-Hoc Ranking

The Ad-Hoc Ranking method of Cavnar and Trenkle [19] relies on the comparison of two models which are ranked in descending frequency (see Figure 1). For every unclassified document, text features are extracted into a document model. In the same way, text features are extracted into a

language model from training data. All features are sorted by their descending frequency (rank). The features contained in the document model are successively searched in the language models.

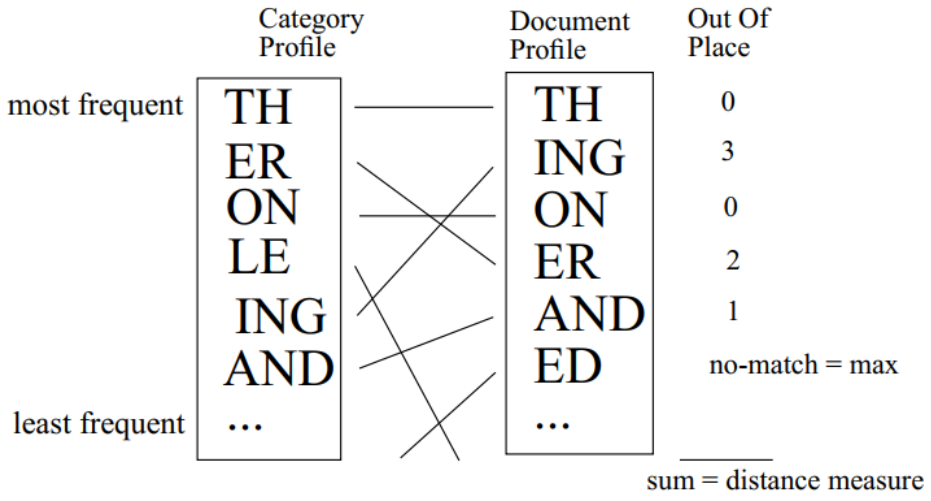

Figure 2.1 Example of the rank-order statistics classifier [19]

First, single out-of-place measure is computed by comparing the entities of the two models and their rank. Then, a value is assigned. The resulting total distance of the out-of-place measures is used for assigning the language to the document. Otherwise, if an entity of the document model is not included in the language model, a maximum value is added to the total distance. This value used for distinguishing the no-matching language from the correct one. Finally, all total distances between the document model and the language models are computed. The language model with the smallest total distance is chosen as describing the language of the document.

2.4.6 Cumulative Frequency Addition

This classifier uses a list of N-grams generated from an input document, without considering their frequencies or doing any sorting of the n-grams. This list can contain duplicates. Each N-gram in this list is searched in the language model of a considered language. The internal frequencies of the N-grams found in the language model are summed up. The bigger the sum is, the smaller is the distance between the language model and the document model. Finally, the language with the maximal sum is chosen as the language of the document. This classification method removes the need to generate a frequency table for each language model, and the computationally intensive sorting of the language model in descending order [8].

Either of the following formulas can be used to calculate the weight of n-gram features based on their frequencies of occurrence in the training language corpus:

$$FI\ (I, J) = C\ (I, J) / \sum I\ C\ (I, J)$$

$$FO\ (I, J) = C\ (I, J) / \sum I, J\ C\ (I, J)$$

FI (I, J) = Internal Frequency of an N-Gram I in Language J

FO (I, J) = Overall Frequency of A N-Gram I in Language J

C (I, J) = Count of the I^{th} N-Gram in the J^{th} Language

$\sum$ I C (I, J) = Sum of the Counts of All the N-Grams in Language J

$\sum$ I, J C (I, J) = Sum of the Counts of All the N-Grams in all the Languages

The internal and overall frequencies of each N-gram usually normalized by dividing each value by the highest frequency of the entire training database and then adding 1 to each value. This makes the final weight of each N-gram frequency normalized to a value between 1 and 2.

The identified language l of a document D is thus determined as the most likely language following Internal Frequency (FI) to weight n-gram frequencies:

$$l = argmax\ Li \in L \sum (C\ (i, j) / \sum i\ C\ (i, j))$$

2.4.7 Bayesian Classification

Language detection is a classification task, Naive Bayes is a classical approach to the problem. A Naive Bayes classifier uses conditional probabilities of observing features in a text to deduce a probability of a text to be written in a given language. In our case the n-grams serve as features [1].

Many classification problems are solved using the Naïve Bayes Classifier. In order to compare the performance of our CFA classification method we have used NBC in our study on the same training and test corpus. A naïve Bayes classifier uses the concept of Bayes' theorem [14]. This classifier assigns the most likely classes to an input string, based on the highest a posteriori probability, given the input string.

For text language identification purpose, a Naïve Bayes classifier can be constructed using n-grams as features. Let T be a set of training samples and let each sample be represented by n

feature vectors, X = x1, x2… xn, with their class labels. Let there be m classes representing the languages: L1, L2….Lm. to predict, a sample Xn is selected to belong to class Li, if and only if:

$$P(Li|X) > P(Lj|X)\ ;\ for\ 1 \leq j \leq n;\ j \neq i$$

Where P(Li|X) is the probability of a class Li given a sample X. Bayes' theorem states that:

$$P(Li|X) = \frac{P(X|Li)P(Li)}{P(X)}$$

Number of the learning algorithms applied to LangID can be understood in the framework of Bayesian classifier, in which it computes *P* (*Li*|*D*), the probability of a given language *Li* from a closed set of candidate languages *L* given a particular document *D*. The identified language *l* of document *D* is thus determined as the most likely language conditioned on the document *D*

$$l = argmax\ Li \in L\ \frac{P(X|Li)P(Li)}{P(X)}$$

2.5 The Ethiopic Writing System

2.5.1 History of Ethiopic

Ethiopia is the home of many ethnics, nations and nationalities. The country has more than 80 languages where Ethiopic and Latin are the widely writing systems. Ethiopic is the writing system for Ethio-Semitic languages we have under study in this research.

Ethiopic as a writing system has been used since the time of early Axumite Empire. Geez as the early language since the time of the Axumite Empire, developed the full set of Ethiopic writing system which Amharic later adopted for its own use.

Being a Semitic language of the Afro-Asiatic language group, Amharic language is related to Hebrew, Arabic, and Syrian. The current Amharic writing system was adopted from the Ge'ez writing system, which was the classical language of the Axum Empire of Northern Ethiopia. It existed between the 1st Century A.D. and the 6th Century A.D. The ancient Sabaean script is in turn attributed as the source of the Ge'ez script. As the Sabean script descended into Ge'ez and later into Amharic, the numbers of symbols in its original Sabean script and their shapes have been changed over time [13].

When the power base of Ethiopia shifted from Axum to Amhara between the 10th and 12th Century A.D., the use of the Amharic language spread its influence, hence became the national language of the country until the advent of Ethiopian People's Revolutionary Democratic Front (EPRDF) to power in 1983 E.C [13].

The use of Amharic as a working language popularized the Ethiopic writing system for use in all the local languages.

Following EPRDF taking leadership, Amharic language continue its usage as the main working language of the Federal Government. Regional governments use Amharic as a working language side by side with their respective local languages.

2.5.2 Current usage of Ethiopic writing system

Ethiopic is a syllabic-alphabetic writing system which has 33 basic characters each having 7 forms or orders for each consonant-vowel combination. Each character occurs in one basic form and in six other forms which is made in accordance with the sound that goes with the symbol [13].

Currently all Ethio-Semitic languages adopted this writing system and extends it by adding new glyphs to it following its rules of writing.

Amharic uses the full array of alphabets, numbers and punctuation marks inherited from the Geez language as a writing system. Following its extensive use as a working language, it dropped the Geez numerals resorting to the Arabic numerals.

The adoption of the Ethiopic in Guragigna and Tigrigna and other local languages as a writing system, provided for a need for new glyphs to extend the writing system to represent the phonemes that are unique characteristics of these languages. Annex I lists the standard and extended character sets of Ethiopic and their UTF encodings.

2.6 Review of related work

2.6.1 Global Researches

Many researches are done on European and Asian languages to test the effectiveness of statistical approaches for language identification. We cannot exhaustively discuss on this researches in this

document. From our review, very few were done on comparative study of the performance of Cumulative Frequency Addition (CFA), Rank-Order Statistics (ROS) and Naïve Bayesian Classifier (NBC).

According to Bashir, Sung-Hyuk and Charles [7] on language identification study, using 12 European languages for training and 5 different languages for testing, using fixed length character n-gram extracted from text without word tokenization, the values of N running between N= (2, 3, 4, 5, 6, and 7), having 81Kb as average corpus size for each language, the performance of classifiers in terms of Accuracy is indicated in Figure 2.3.

Based on the comparative performance of these classifiers presented in Figure 2.3 CFA performs better for shorter strings and maintains its performance in par with ROS as the test strings grows larger and larger. NBC performs lower in all ranges of test phrase lengths as compared to CFA and ROS.

The strength of CFA and NBC are their speed of classification as compared to ROS, for ROS require sorting of n-gram frequencies in the language model for later classification task. We can achieve the performance of ROS classifier and avert the need for sorting by using CFA as a classifier [7].

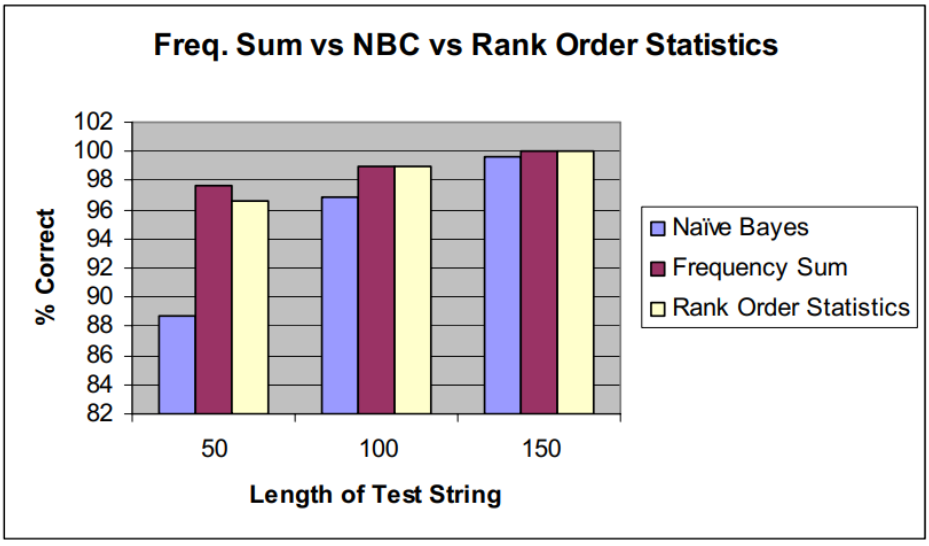

Figure 2.2 Percentage accuracy of classification of NBC, CFA, and ROC statistics [7]

According to Leonid Panich [8], comparative study of language CFA, NBC and ROS classifiers on Spanish, Catalan, Basque and Galician languages using Twitter posts the performance of CFA

classification method is studied and compared with NBC and ROS. Tests were conducted on the effectiveness of the methods on both short and long text documents. The CFA outperforms the NBC classifier on Non-Latin data sets. Arabic, Russian, Hindi, and Ethiopic are fewer examples of Non-Latin writing systems.

According to Dunning [12] on his research on English and Spanish language using NBC classifier, he omitted the tokenization step for building the N-grams with window running between 1 and 7. He used training sets of the length between 1,000 and 50000 bytes for the English and Spanish language and test text of length from 10 to 500 bytes. He stated, that classification accuracy is good and improves with longer test strings and with more test data. With shorter test strings or with less training data, bigram models appear to perform best [12]. His approach worked with an accuracy of 92% with 20 bytes (characters) of validation data and 50,000 characters of training whereas reached 99.9% accuracy for 500 characters test text on the same amount of training set. With validation strings of 500 bytes and training text of 5,000 characters he obtained an accuracy of 97%.

2.6.2 Local Research

The first research on the automatic language identification is done by Kidist [6]. It is based on a hybrid approach that combines Rule based with Machine Learning to create a general purpose language identifier to effectively identify Ethio-Semitic languages both on short and long documents written using Ethiopic. The researcher used character n-grams to develop the language models, and Naïve Bayes as classification method to perform similarity measures, while Precision, Recall and F-measure used as performance measures. An average of 559,000 words is used for each language in the corpus after data cleaning that was further divided in to training and test sets [6].

The research by Kidist [6] is worth mentioning as it is the first of its kind on language identification studies on Ethio-Semitic languages. To our knowledge, there are no similar researches done on Ethio-Semitic languages using any of the available modelling and classification techniques for Language Identification purpose. In an attempt to improve the performance of the classifier, two types of CBN language models were tested with a two variation made considering and omitting the location features. The test results of these models is summarized in Table 2.2 and 2.3.

N-gram	Amharic		Geez		Guragigna		Tigrigna	
	Without Location	With Location	Without Location	With Location	Without Location	With Location	Without Location	With Location
2-gram	66.25%	67.37%	79.30%	76.87%	62.93%	69.49%	76.04%	75.16%
3-gram	65.59%	68.20%	79.72%	74.91%	67.49%	70.31%	75.62%	76.92%
4-gram	67.74%	70.39%	80.59%	76.95%	68.33%	73.62%	76.82%	77.57%
5-gram	68.99%	70.38%	78.24%	79.15%	70.27%	73.69%	77.11%	78.98%

Table 2.2 The F-Score of fixed length n-grams using NBC [6]

The performance of fixed length character N-grams with and without location features as summarized in Table 2.2 above, as highlighted in green, the maximum performance of 80.59% is achieved on 4-grams without location features and 78.98% with location features on 5-grams.

The tests on infinity n-grams with and without location features is summarized as follows on Table 2.3

10-Fold cross validation	Amharic		Geez		Guragigna		Tigrigna	
	Without Location	With Location	Without Location	With Location	Without Location	With Location	Without Location	With Location
Average	83.07%	85.96%	82.95%	86.26%	86.67%	88.25%	85.53%	88.75%

Table 2.3 The F-Score of Infinity n-gram using NBC [6]

The maximum F-score achieved for NBC classifier on Infinity n-gram based language model is 88.75% and 86.67% with and without considering location feature respectively as highlighted in green on Table 2.3.

Our research considers both fixed and infinity n-gram language models tested by Kidist [6] and the fixed length character n-grams generated from text without tokenization approach used by Dunning [12] to perform a comparative study of CFA and NBC classifiers.

CHAPTER THREE

DESIGN AND IMPLEMENTATION OF MODELS AND CLASSIFIERS

3.1 Introduction

For the purpose of evaluating the CFA against the NBC classification method, we have built different language models based on the experiment conditions listed under section 1.7.6 and performed our evaluation under two different evaluation contexts indicate under section 4.5.

Corpus and output files are saved as text files with .txt extension. Comma, "," is used as delimiter for all output files such as model, testing samples, test results. The corpus and complete project files are publicly available and can be downloaded from https://github.com/Rediat/cfa.git.

3.2 Design Goals

The design aim is to build a system that:

1. Partition that takes source corpus, clean and generate 10 different Training and Testing corpora

2. Generate 5 different ways of generating character based n-gram language models from each of the 10 Training corpora. A total of 50 (i.e. 5x10) language models are generated.

3. Generate 10 testing documents with test phrases length running from 1 to 25 words from each of the 10 testing corpora.

4. Perform classifications using CFA and NBC classifiers over each of the language models and testing phrases and save the results to text files.

5. Generate the average performance results in tabular and graphical presentations.

3.3 Architecture of the system

Our system is designed following the architecture depicted in Figure 3.1

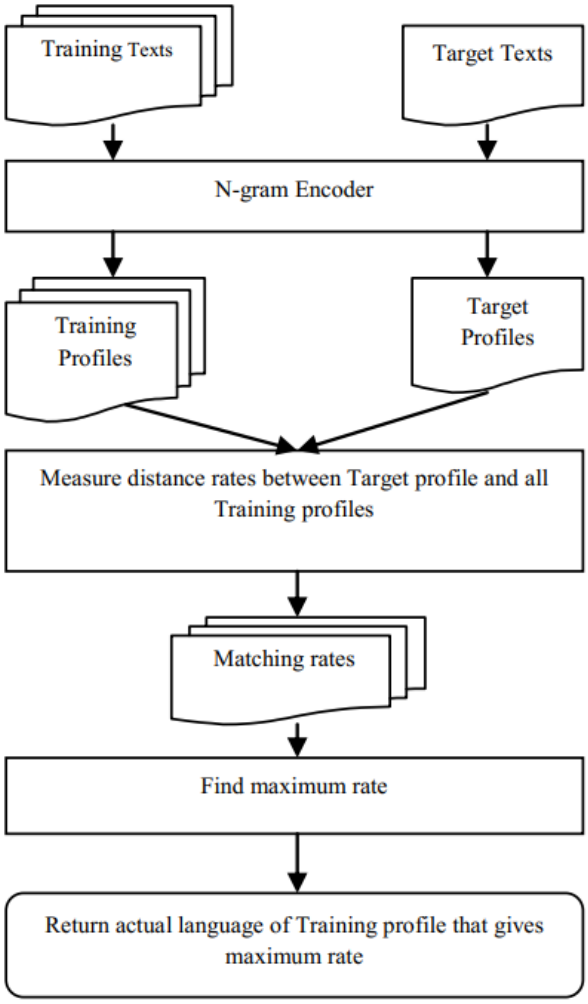

Figure 3.1 System flow of language identification process for both CFA and NBC

3.4 Data Source

The data source for building the language profiles and developing a testing set is the Online Christian Bible written in Amharic, Geez, Guragigna, and Tigrigna text as indicated in section 1.6.2. The advantage of using the Bible is that it is a source for a well curated text. For lack of access to the digital bible in Geez, we have used the Old Testament version whereas using the New Testament contents for other languages. Each language document is represented both in machine readable and human interpretable form in text files using UTF-8 encoding. Table 3.1 details the word distributions in the corpus after data cleaning.

3.5 Data Preprocessing

The data cleaning for the testing and training corpus is done through removal of punctuation and other special marks, Arabic numbers, English characters, tabs and extra white spaces. These features are removed to maintain the unique features relevant for document representation.

Data cleaning is the process of filtering of the corpus to maintain only relevant features for language modelling. We have removed features such as Numbers, Punctuation marks, mathematical symbols, Arabic numbers, English characters, special characters and multiple white spaces since they are not much help in differentiating among languages. This reduce the corpus size by filtering and maintaining only the relevant features for language modelling.

In Geez, we have maintained the numbers as relevant feature to the language for they are written using the Geez numerals. Amharic, Guragigna and Tigrigna use the Arabic numerals instead of the Geez numerals. We have excluded the numbers from the language models as non-relevant feature to assist in language identification.

We have tokenized the source text in to words using the white space as a marker for converting the stream of text in to BOW for each language.

Word tokenization is performed for models developed based on n-grams generated from BOW, such as fixed length n-grams and Infinity n-grams.

As highlighted in green, after data cleaning based on Table 3.1, the most frequent word is a 3 character words which consists of 26.46% of the Amharic corpus and 29.68% of the corpora. 90.44% of the entire corpus consists of words with maximum length of 5 characters or less (i.e. words with 1, 2, 3, 4 or 5 characters long) as highlighted in blue on Table 3.1.

Length in		Amharic		Geez		Guragigna		Tigrigna		Total	
Characters[2]	**Bytes[3]**	**%[4]**	**Cum. %[5]**	**%**	**Cum. %**	**%**	**Cum. %**	**%**	**Cum. %**	**%**	**Cum. %**
1	76	0.22%	0.22%	0.06%	0.06%	0.13%	0.13%	0.15%	0.15%	0.14%	0.14%
2	78	16.45%	16.67%	11.82%	11.88%	22.52%	22.66%	27.94%	28.09%	19.90%	20.03%
3	80	26.46%	43.13%	32.70%	44.58%	32.51%	55.16%	27.00%	55.09%	29.68%	49.72%
4	82	25.25%	68.38%	29.08%	73.66%	24.06%	79.22%	24.15%	79.24%	25.60%	75.32%
5	84	16.42%	84.79%	16.80%	90.45%	14.62%	93.84%	12.90%	92.14%	15.12%	90.44%
6	86	9.31%	94.10%	5.27%	95.73%	4.65%	98.49%	4.99%	97.13%	5.98%	96.42%
7	88	3.61%	97.70%	3.71%	99.44%	1.36%	99.85%	2.00%	99.13%	2.64%	99.06%
8	90	1.72%	99.42%	0.49%	99.93%	0.11%	99.96%	0.67%	99.80%	0.73%	99.78%
9	92	0.47%	99.89%	0.06%	99.99%	0.04%	100.00%	0.15%	99.95%	0.17%	99.96%
10	94	0.09%	99.99%	0.01%	100.00%	0.00%	100.00%	0.04%	99.99%	0.04%	99.99%
11	96	0.01%	100.00%	0.00%	100.00%	0.00%	100.00%	0.01%	100.00%	0.01%	100.00%
Total		**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Counts[6]		**7,610**		**8,090**		**8,286**		**8,679**		**32,665**	

Table 3.1 The Distribution of words in the language corpus

[2] Word length measured by the number of characters
[3] The average size of a given character word in bytes
[4] The percentage of words with length in characters
[5] Cumulative relative frequency of words by adding the previous rows in 9
[6] The total counts of words out of which the percentages in 9 are computed. We can calculate the number of words 3 character long in the Amharic corpus by multiplying 26.46% * 7,610 which is 2,014 words.

3.6 Training and Testing Corpus

We divided each of the languages corpus indicted on Table 3.1 into 90% for training and 10% for testing purpose using ten-fold cross validation technique.

Table 3.2 shows the distribution of the average training and testing corpus in words across languages in each testing rounds.

Languages	Training	Testing
Amharic	7,610	846
Geez	8,090	899
Guragigna	8,286	921
Tigrigna	8,679	964
Total	**32,665**	**3,629**

Table 3.2 Training and testing corpus size in word counts for each language

3.7 Generating Language Models

Modeller is the component of the designed system used to generate the language profiles based on the required testing conditions listed under section 1.6.4.

3.7.1 N-grams generated from BOW

This is a modelling technique used to generate n-grams after data cleaning and word tokenization of the training corpus. Since this representation ignores the order or position of words in the text, it is called a Bag of Words (BOW). There are two ways of extracting n-gram features from the BOW. These are Fixed Length Character N-grams and Variable Length character n-grams. These can be generated in two variations; with or without considering the location features.

1. **Fixed Length Character N-grams**

This is a text representation technique which uses a fixed list of n-grams as (N = 2, 3, 4 and 5) in which a text document is represented with character n-grams for each values of N.

The feature extraction based on fixed length n-grams is used for both training and testing strings. For example the Amharic word "እግዚአብሔር", which is extracted from given testing document can be represented with the following character n-gram with fixed size trigram, N = 3.

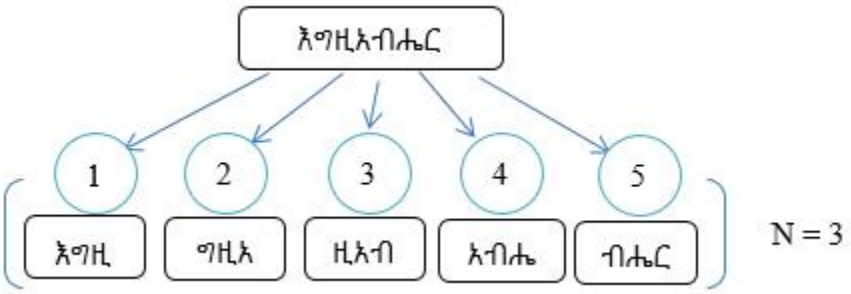

Figure 3.2 Character trigram representation of the Amharic word "እግዚአብሔር"

Similarly we have generated for the full set of N=2 up to N=5 to extract the features and build a language model for each language.

2. Variable Length character n-grams

Fixed length character n-grams generate relatively limited set of features for the language classifier as larger values of N beyond the max fixed size will not be considered for modelling and classifications. To extract more features for language modelling, considering the full set of n-grams where n=2, 3, … len(w), where w is a word and len(w) is the number of characters in the given word, will provide for a richer set of n-grams.

Variable length n-gram depicted in Figure 3.3 is "All Substring Features" according to the Daisuke [15]. This technique is considered character n-grams from BOW with N=1 … ∞ (where ∞ stands for the maximum number of characters in a word) and hence referred as "Infinity N-grams" according to Kidist [6].

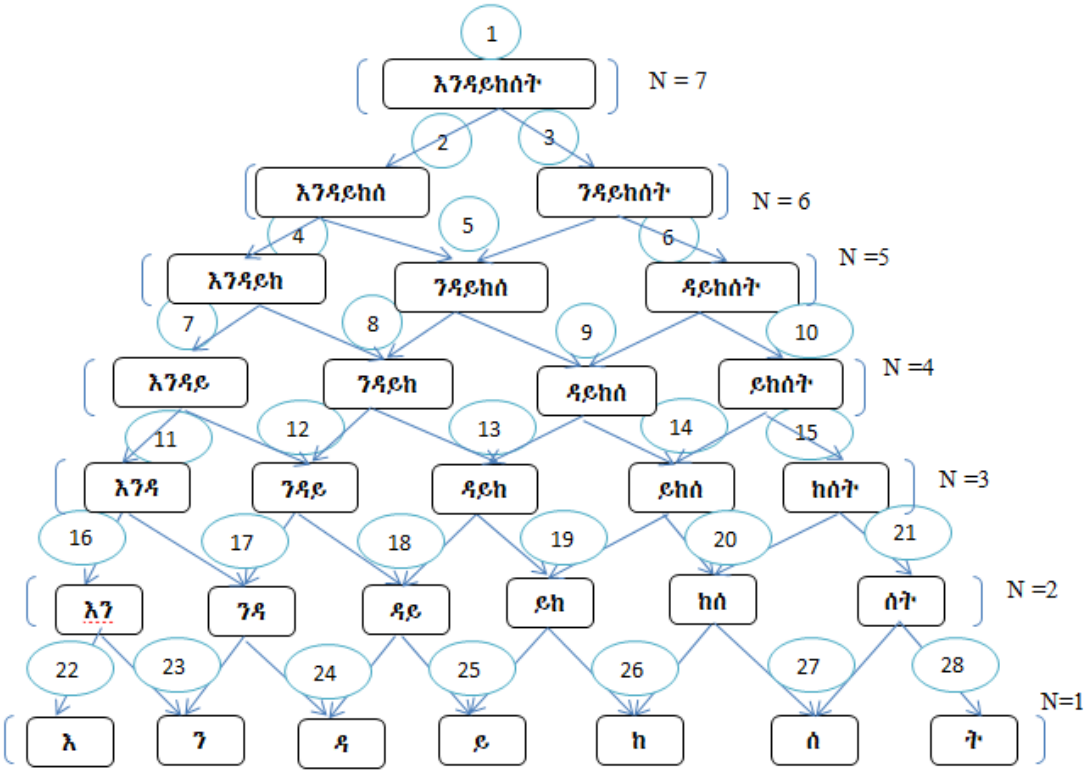

Figure 3.3 Infinity N-grams for the word "እንዳይከሰት" [6]

We have taken the minimum value of N=2, excluding N=1, and let the maximum value of n varies to match the maximum number of characters in a word from which the n-gram features were extracted.

3. N-gram Generated location features/markers

Once the text from the training corpus is broken down in to BOW though the process of word tokenization we have added N-1 underscores on both sides of the word, where N is the N-gram i.e. 2, 3, 4, etc. considered at a time. This helps to mark the deepest characters in addition to the start and end N-grams. Location markers at the beginning and ending of a word helps to make the distinction between the outer and deeper-word N-grams. From the word "እውነት" the following N-grams are generated for fixed length and infinity n-grams:

Bi-grams: _እ, እው, ውነ, ነት, ት_

Tri-grams: _ _እ, _እው, እውነ, ውነት, ነት_, ት_ _

Quad-grams: _ _ _እ, _ _እው, _እውነ, እውነት, ውነት_, ነት_ _, ት_ _ _

Pent-grams: _ _ _ _እ, _ _ _እው, _ _እውነ, _እውነት, እውነት_, ውነት_ _, ነት_ _ _, ት_ _ _ _

This will help us generate more features and allow us to mark even the deeper characters in a word, apart from marking only the beginning and end of a word.

3.7.2 N-grams generated from string of source text

This is the n-gram model generated according to Dunning [12] considering the whole source text to be broken down into n-grams. Some authors also reported replacing space character with "_" (underscore), but similar to Bashir et.al. [7]. We maintained the white space in between the text as a valid character. As generating n-grams from BOW loses much of the document information, this approach helps to capture more features across word boundaries from the training corpus.

From the word "እውነት እላችኋለሁ", including the white space as valid character, we can extract all possible tri-grams as "እውነ", "ውነት", "ነት ", "ት እ", " እላ", "እላች", "ላችኋ", "ችኋለ", "ኋለሁ".

3.7.3 Distribution of N-grams across language models

We can generate 5 different types of language models based on fixed length and Infinity n-grams with and without considering the location markers. These models are listed as experiment items under section 1.6.4. The distribution of N-grams generated using the modelling techniques is summarized in Table 3.3 as follows:

N-grams	Average N-gram Vocabulary under each Language Models					
	BL[7]	**BY[8]**	**FL[9]**	**IL[10]**	**IN[11]**	**Total**
2	12,616	13,764	13,766	13,691	12,610	66,447
3	18,971	37,688	30,453	28,813	18,976	134,901
4	14,598	72,262	45,600	36,016	14,598	183,073
5	7,713	97,009	59,126	28,856	7,713	200,417
6	0	0	0	16,641	3,152	19,793
7	0	0	0	7,349	1,060	8,409
8	0	0	0	2,835	312	3,147
9	0	0	0	908	78	987
10	0	0	0	231	15	246
11	0	0	0	38	2	40
Total	**53,898**	**220,723**	**148,946**	**135,378**	**58,516**	617,461

Table 3.3 The distribution of n-grams across language models

[7] BL - Fixed length N-grams generated from BOW without location features
[8] BY - Fixed length N-grams generated from text without word tokenization
[9] FL - Fixed length N-grams generated from BOW with location features
[10] IL - Infinity N-gram generated from BOW with location features
[11] IN - Infinity N-gram generated from BOW without location features

The average of the 10 different training set is considered to calculate the distribution of n-gram vocabularies in Table 3.4. The longest word in the training corpora has 11 characters. Having the baseline model (BL) as a reference point, we have 4, 2.76, 2.51, and 1.1 times the size of the baseline vocabularies in language models BY, FL, IL and IN respectively.

3.7.4 N-gram Frequency Calculation

The modelling component our system calculated the total N-gram counts for each language as well as the overall N-gram count for the entire training set. This helps to calculate the conditional probabilities using the NBC classifier during classification. For CFA we have calculated the weights of each N-grams features during modelling using overall frequency.

Table 3.4 shows a sample calculations for N-gram "አብ".

Language	N-gram	N-gram Counts	Total N-grams in this Language	Total N-grams in all Languages	Internal Frequency	Overall Frequency
Amharic	አብ	116	106,977	443,223	116/106,977	116/443,223
Geez	አብ	153	113,997	443,223	153/113,997	153/443,223
Guragigna	አብ	48	112,494	443,223	48/112,494	48/443,223
Tigrigna	አብ	78	109,751	443,223	78/109,751	78/443,223

Table 3.4 A sample of how internal and overall frequencies are calculated

As in the case of the Overall Frequency calculation in Table 3.4 the relative frequency calculation 116/443,223 = 2.617192699837327936501490220498 5e-4 yields very small floating point number. Without affecting the classifier's performance, to bring this to a human understandable form for comparisons, the internal and overall frequencies of each N-gram were normalized by dividing each value by the highest frequency of the entire training database and then adding 1 to each value. Thus, the final value of each N-gram frequency was normalized to a value between 1 and 2 for ease of comparison as indicated in Table 3.5 [7]. For no particular reason, as we have selected the Overall Frequency for our classifier.

Language Tag	N-gram	Characters	Frequency	Bytes	Overall Frequency
ti	ድማ	2	588	78	1.0000002350542088
ti	እማዚአ	4	153	82	1.0000000611620645
gu	ባረ	2	553	78	1.0000002210628869
gu	ይእማዘር	5	186	84	1.0000000743538824
ge	ይቤሎ	3	368	80	1.0000001471087565

Table 3.5 A sample from fixed length n-gram without location feature language model

3.8 Test String Generations

This component reads each file from the testing corpus, identify the positons of the white spaces for generating unilingual word based n-grams as test phrases with length between 1 to 25 words. The test phrases were saved in 10 different text files corresponding to their source training corpus, as comma delimited text files.

Example of breaking a phrase "ለመከሩ ሠራተኛን ይጨምር ዘንድ" extracted from one of the testing corpus generates a combination of the strings in Table 3.6.

Language Tag	Testing N-gram	Words	Characters	Bytes
am	ለመከሩ	1	4	82
am	ለመከሩ ሠራተኛን	2	10	94
am	ሠራተኛን	1	5	84
am	ለመከሩ ሠራተኛን ይጨምር	3	15	104
am	ሠራተኛን ይጨምር	2	10	94
am	ይጨምር	1	4	82
am	ለመከሩ ሠራተኛን ይጨምር ዘንድ	4	19	112
am	ሠራተኛን ይጨምር ዘንድ	3	14	102
am	ይጨምር ዘንድ	2	8	90
am	ዘንድ	1	3	80

Table 3.6 A sample of the testing phrases generated from a test string.

3.9 Classification

The classifier component reads the language models, generate the n-grams for the test phrases in a similar way the language model generated, performs similarity measures for each classification task, and save the performance measurements in to text files.

3.9.1 Classification using Cumulative Frequency Addition (CFA)

The test strings will be broken down in to n-grams of the same set of window sizes as we did for modelling. We generate the n-gram of each test strings with the selected phrase lengths similar to the n-grams generated for the language model considered for classification. The advantage of using CFA classification method is the fact that unlike NBC Classifier, it does not need a smoothing factor. The probabilities are added cumulatively and adding 0 for the missing features will not affect the classifier for the missing n-grams in the training set. NBC assumes that the probabilities of unique N-grams are independent of each other, and this is not a good assumption to begin with for language modelling. For example, the occurrence of "th" increases

the probability that the word may be “the” or ‘that.” Thus, assuming that “th” is independent of “that” or “the” is not a valid assumption. This may explain why CFA performs better than NBC on short test strings [7].

3.9.2 Classification using Naïve Bayes Classifier (NBC)

Similar to the CFA classifier above the test strings are broken down to match the language model n-grams. Laplace’s rule of succession recommends to add an additive smoothing λ = 1 to the counts of all possible n-grams. This is to maintain the products of probabilities from becoming zero for few missing character n-grams from the training models. This is formally stated as follows:

$$P_{add}\left(Xi\middle|X_{i-n+1}^{i-1}\right) = \frac{C\left(X_{i-n+1}^{i}\right) + \lambda}{C\left(X_{i-n+1}^{i-1}\right) + \lambda V'}$$

Where xi, i denotes the n-gram xi . . . xj, V is the size of the vocabulary (counts of all unique characters n-grams across all languages) and C(x) denotes the count of occurrences of an n-gram in a given language. We have used the Laplace additive smoothing with λ = 1 for all n-grams in a test string.

The classifier reads the relevant model selected for the test round with testing strings from the samples folder, and do the classification task.

It generates the output of the classification measurements of Accuracy, Precision, Recall and F-score for each classification tasks.

3.10 Empirical Evaluation

Language Identification is the Text Classification problem where, given a set of evaluation documents, each having a known correct label from a closed set of labels, language annotations in our case, and a predicted label for each document from the same set of annotations.

Language level accuracy is the proportion of documents that are correctly labelled over the entire evaluation collection. This is the most often-reported metric, and conveys the same information as the error rate, which is simply the proportion of documents that are incorrectly labelled (i.e. 1 - accuracy).

There are two distinct ways in which results are generally summarized: (1) precision, in which documents are grouped according to their predicted language; and (2) recall, in which documents are grouped according to what language they are actually written in.

More formally, consider a set of documents $D = \{D1 \dots Dm\}$ and a set of languages $L = \{L1 \dots Ln\}$. For each document *Dx* we denote that the document is written in language *Ly* by *Dx* $\rightarrow Ly$, and that the system predicts the document is written in Lz by $Dx \rhd Lz$. We use an over line to denote negation, for example $Dx \rhd \overline{Ly}$ denotes that Dx is not written in Ly. For each language $Li \in L$, each document can fall into four possible categories:

True Positive (TP) $Dx \rightarrow Li \; and \; Dx \rhd Li$

False Positive (FP) $Dx \rightarrow \overline{Li} \; and \; Dx \rhd Li$

False Negative (FN) $Dx \rightarrow Li \; and \; Dx \rhd \overline{Li}$

True Negative (TN) $Dx \rightarrow \overline{Li} \; and \; Dx \rhd \overline{Li}$

The Confusion Metrix

A confusion matrix, also known as an Error Matrix is a specific table layout that allows visualization of the performance of a classifier. Each row of the matrix represents the instances in a predicted class while each column represents the instances in an actual class (or vice versa). But in our case the row represent the actual class and the column represents a predicted class. This is a preparation to tabulate the performance of the classifier for latter computation of the Precision and Recall. Once these measures are calculated, they will be used for computing F-score with (alpha=1 i.e. giving equal weight for precision and recall).

Table 3.7 indicates that for cases where a document from Amharic (am) test corpus is classified as Geez (ge), Guragigna (gu) or Tigrigna (ti), the counts of this number of wrong classifications are placed in the corresponding language cells highlighted in yellow as False Negative (FN) and the counts where it is correctly classified as Amharic is indicated as True Positive (TP) in the cell indicated with green. The counts of wrong predictions of other languages as Amharic (am) is a False Positive and stated under predictions to Amharic corresponding to the actual language of the documents as highlighted in blue in Table 3.7. True Negatives (TN) are the counts of predictions that non Amharic (am) documents are classified correctly as non-Amharic but need

to be correctly labelled as TP, FP and FN corresponding actual and predicted language classes accordingly.

From the Amharic (am) testing documents Table 3.7 indicates the TP, FP, and FN used to calculate Precision and Recall.

		Predicted Class				**Total**
Actual Class		am	ge	gu	ti	
	am	TP	FN	FN	FN	TP+FN
	ge	FP	TN	TN	TN	
	gu	FP	TN	TN	TN	
	ti	FP	TN	TN	TN	
Total		TP+FP				

Table 3.7 The Confusion Matrix for evaluating classifier performance

The frequency of each category can be tabulated for each classifier across languages basis on these counts, Precision (*P*), Recall (*R*) and F-score (F) are defined as the following ratio of counts:

$$P = \frac{TP}{TP + FP} \quad R = \frac{TP}{TP + FN} \quad F = 2.\frac{P.R}{P + R}$$

Table 3.8 below indicates the actual counts made from the classification to calculate Precision and Recall.

		Predicted Class				**Total**
Actual Class		am	ge	gu	ti	
	am	654	10	23	3	690
	ge	8	758	0	0	766
	gu	0	0	844	4	848
	ti	18	88	4	687	797
Total		680	856	871	694	3,101

Table 3.8 Actual counts made from the classification to calculate the Precision, Recall.

Individual Precision [Vertical] for the four languages calculated based as $P = \frac{TP}{TP+FP}$ where the languages Amharic, Geez, Guragigna and Tigrigna indicated as am, ge, gu and ti respectively.

$$P(am) = \frac{654}{680} = 0.96, P(ge) = \frac{758}{856} = 0.89, P(gu) = \frac{844}{871} = 0.97, P(ti) = \frac{687}{694} = 0.99$$

Based on this individual Precision values we can calculate the Average Precision of a classifier across languages as follows.

$$Average\ P = \frac{0.96\ +\ 0.89\ +\ 0.97 + 0.99}{4} = 0.95$$

Individual Recall [Horizontal] for the four languages calculated based as $R = \frac{TP}{TP+FN}$ where the languages Amharic, Geez, Guragigna and Tigrigna indicated as am, ge, gu and ti respectively.

$$R(am) = \frac{654}{690} = 0.95, R(ge) = \frac{758}{766} = 0.99, R(gu) = \frac{844}{848} = 0.99, R(ti) = \frac{687}{797} = 0.86$$

Based on this individual Recall values we can calculate the Average Recall of a classifier across languages as follows.

$$Average\ R = \frac{0.95\ +\ 0.99\ +\ 0.99 + 0.86}{4} = 0.95$$

From individual values of Precision and Recall we calculate the micro F-score as follows

$$F(am) = 2\frac{0.96 * 0.95}{0.96 + 0.95} = 0.95, \qquad F(ge) = 2\frac{0.89 * 0.99}{0.89 + 0.99} = 0.89$$

$$F(gu) = 2\frac{0.97 * 0.99}{0.97 + 0.99} = 0.98, \qquad F(ti) = 2\frac{0.99 * 0.96}{0.99 + 0.96} = 0.97$$

Based on this individual F-score values we can calculate the Average F-score of a classifier across languages as follows.

$$Average\ F = \frac{0.95\ +\ 0.89\ +\ 0.98 + 0.97}{4} = 0.95$$

3.11 Measuring Classification Performance

The tester is designed on top of the Classifier that computes the performance metrics for each classifier on each language model and evaluation contexts in section 3.9. It performs iterative tests by changing the test phrase length in 1, 2, 3, 4, 5, 10, 15, 20, and 25 to take measurements of Accuracy, Precision, Recall and F-score over varying test phrase length over the 10 different training and testing corpuses and saved the results of both classifiers in to result files for use by

the plotter. The result files used also to build Tabular presentation of the classification performances.

3.12 Graphical Presentation of Performance Metrics

The plotter takes the output files of the Tester for each testing conditions and generates the graphical presentations depicted on from Figure 4.1 to 4.26 for performance metrics, Accuracy, Precision, Recall and F-score.

CHAPTER FOUR

EXPERIMENTATION AND ANALYSIS

4.1 Introduction

Our aim is to measure the performance of CFA classifier on Ethio-Semitic languages and do a comparative analysis with the most widely used NBC classifiers' performance generated from the same training and test corpus. The experiment is conducted under two different evaluation contexts to identify a classifier, the evaluation context and the language model that yields the highest performance in terms of F-score.

We have used Precision, Recall and F-score to measure the individual performance of classifiers on each language models in general, and F-score to make comparisons across classifiers, language models and evaluation context. F-score provides for an optimal measure by taking both Precision and Recall in to account.

Precision is the proportion of correctly classified test samples in all samples classified to the given language, whereas Recall is the proportion of correctly classified test samples in all samples of the given language as indicated in Section 3.9. In this chapter, the experimental setup and the detail experiments conducted for the research are described.

4.2 Evaluation Context

We have considered two different evaluation context for our experiments.

Context I – In this context we are considering all n-grams from N=2 all at once and measure the classifiers' performances. For example, the one word test phrase "እውነት" will perform language identification considering all the possible n-grams (up to N=5 for fixed length, and N up to the number of characters in a word for infinity n-grams), that can be generated from the phrase < [እው, 2], [ውነ, 2], [ነት, 2], [እውነ, 3], [ውነት, 3], [እውነት, 4] >. In this investigation, we explore the language models and identify the one that will provide for the maximum classifiers' performance on both short and long test phrases.

Context II – This is the most commonly used comparison of fixed length n-grams performance. For example, the one word test phrase "እውነት" we perform language identification for each

individual values of N considering < [እው, 2], [ውነ, 2], [ነት, 2]> for 2-grams, < [እውነ, 3], [ውነት, 3] > for 3-grams, and < [እውነት, 4] > for 4-grams. In this evaluation context the goal of the experiment is to measure the classification performance of each classifier for each value of N where N runs from 2 up to 5 for fixed length, and N up to the number of characters in a word for infinity n-grams], then select the specific value of N with the highest classification performance.

4.3 Experiment Setup

We have performed a preliminary study on the original corpus by performing the following:

1. Removing common words shared between the languages if that can improve the classifiers performances.

2. Considering language modelling techniques based on short words method from 2 to 5 characters length,

3. Evaluating the most frequent words modelling approach considering 10%, 15%, 25%, 30% and 50% of most frequent words extracted from the training corpus.

4. Excluding of n-gram features from the language models that occurred only once.

The results of our preliminary experiment indicated that, the performance of classifiers deteriorates significantly below 80% applying either of the CBN language modelling approaches listed above. This is because developing language models based on item 1 to 3 and any withhold from the language mode as in item 4 above increases the counts missing n-grams and hence deteriorates the classifiers performance. Our experiments without considering either of the items above showed relatively better performance. Since we are looking for language models and classification methods that can improve the performance of classifiers, we revered to character based n-gram language modelling options that considers all words from the training documents, and classification using any occurrences of n-grams in the language model.

We have chosen to conduct experiments and measure the performance of CFA and NBC classifier over 5 different language models under two evaluation contexts by changing the length of test strings. The performance of the classifiers measured in F-scores goes beyond 98.66% for phrase lengths of 5 words and above.

We have selected 1, 2, 3, 4, 5, 10, 15, 20 and 25 word lengths for test phrases emphasizing our measurement on phrases with 1 to 5 words to measure the power of our classifiers measured in F-Score.

The test strings generated from the testing corpus for testing our classifiers performance has the following distribution detailed in Table 4.1

Words in Phrases	Average number of characters	Average size in bytes	Average number of test phrases			
			Amharic	Geez	Guragigna	Tigrigna
1	4	82	564	524	550	511
2	8	91	784	823	861	883
3	13	100	826	876	904	941
4	18	110	836	887	913	953
5	22	119	839	891	915	957
10	46	166	836	890	912	956
15	69	213	832	885	907	951
20	93	260	827	880	902	946
25	116	306	822	875	897	941

Table 4.1 The Average distribution of Test strings across languages

We have performed the following experiments under each evaluation context over different language models:

1. Exp-1 – Language model generated from fixed length character n-grams taken from a bag of words collected from language training and testing corpus. This is used as a baseline for CFA and NBC classifiers to study how the changes to the model will affect the classifiers' performance. (BL)
2. Exp-2 - N-grams generated from text strings taken from the training corpus (BY)
3. Exp-3 – Fixed Length N-grams generated from BOW with location features. This language model is developed applying the location feature on the language model under Exp-1 (FL)
4. Exp-4 - N-grams generated from Infinity character N-grams taken from a BOW without location features. The value of N begins from 2. (IN)
5. Exp-5 - N-grams generated from Infinity character N- taken from a BOW with location features. This language model is developed applying the location feature on the language model under Exp-4. The value of N begins from 2. (IL)

4.4 Cross-validation

Cross-validation is a technique to evaluate predictive models by partitioning the original sample into a training set to train the model, and a test set to evaluate it.

The common form of cross-validation is k-fold cross-validation. In k-fold cross-validation, the original sample is randomly partitioned into k equal size subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k-1 subsamples are used as training data. The cross-validation process is then repeated k times (the folds), with each of the k subsamples used exactly once as the validation data. The k results from the folds can then be averaged (or otherwise combined) to produce a single estimation. The advantage of this method is that all observations are used for both training and validation, and each observation is used for validation exactly once. In most classification problems 10-fold cross-validation (when k = 10) is the most common used.

We have divided our corpus in to 10 different folds and each $1/10^{th}$ is used as a test corpus and the remaining $9/10^{th}$ for training. Using this procedure we have generated 10 mutually exclusive training and test sets. The averages of each of the measures, Precision, Recall and F-scores, is calculated over the 10 rounds for each classifiers.

4.5 Observations in Experiments

Our experiment is conducted to test the classifiers' performance by changing the training models similarly under each evaluation contexts. Starting from the baseline performance of our classifier in Exp-1 we have studied the changes in performance when the training model changes in the upcoming experiments. A total of 3,049,410 and 19,516,224 observations were made under Context I and Context II respectively across different language models for both classifiers as summarized on Table 4.5 below over the total of 10-fold cross validations.

Words in test Phrases	Context I		Context II		Totals
	CFA	NBC	CFA	NBC	
1	107,430	107,430	687,552	687,552	1,589,965
2	167,530	167,530	1,072,192	1,072,192	2,479,446
3	177,355	177,355	1,135,072	1,135,072	2,624,857
4	179,445	179,445	1,148,448	1,148,448	2,655,790
5	180,050	180,050	1,152,320	1,152,320	2,664,745
10	179,690	179,690	1,150,016	1,150,016	2,659,422
15	178,735	178,735	1,143,904	1,143,904	2,645,293
20	177,735	177,735	1,137,504	1,137,504	2,630,498
25	176,735	176,735	1,131,104	1,131,104	2,615,703
Totals	**1,524,705**	**1,524,705**	**9,758,112**	**9,758,112**	**22,565,719**

Table 4.2 The number of test observations under each evaluation context and classifiers

Table 4.3 summarizes the total number of observations conducted to calculate the averages of metrics across the 10-fold cross validation under each context and each language model.

Words in test Phrases	Context I		Context II[12]		Totals
	CFA	NBC	CFA	NBC	
1	21,486	21,486	34,378	34,378	111,728
2	33,506	33,506	53,610	53,610	174,233
3	35,471	35,471	56,754	56,754	184,452
4	35,889	35,889	57,422	57,422	186,627
5	36,010	36,010	57,616	57,616	187,257
10	35,938	35,938	57,501	57,501	186,888
15	35,747	35,747	57,195	57,195	185,899
20	35,547	35,547	56,875	56,875	184,864
25	35,347	35,347	56,555	56,555	183,829
Totals	**304,941**	**304,941**	**487,906**	**487,906**	**1,585,778**

Table 4.3 The total number test observations from which the average matrices are computed

4.5.1 Exp-1 – Fixed Length CBN without Location Features from BOW – Baseline (BL)

In this experiment, we aim to set a baseline performance for the classifiers, CFA and NBC. By changing the input models in the coming experiments, we can study the factors that affect classifiers' performances. Here, the Language Model is generated from fixed length character n-grams (2, 3, 4, and 5) without considering location features generated from a bag of words extracted from the training corpus.

[12] For each values of N=2,3,4 and 5 excluding the N-grams above 5 for Infinity n-grams

Context I

The Figures 4.1 and 4.2 are the graphical presentations of the performance metrics for both classifiers.

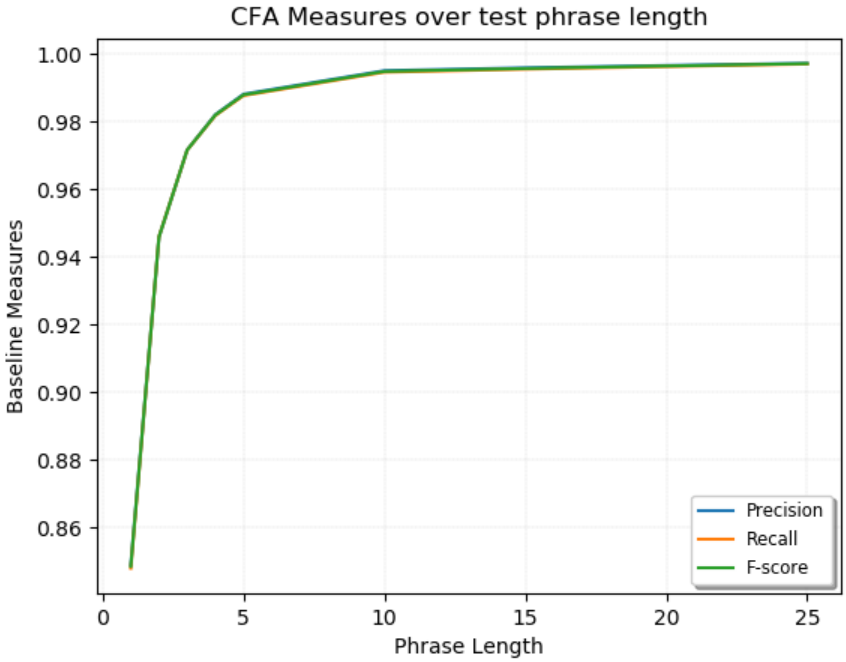

Figure 4.1 CFA matrices on Baseline language model considering all n-grams

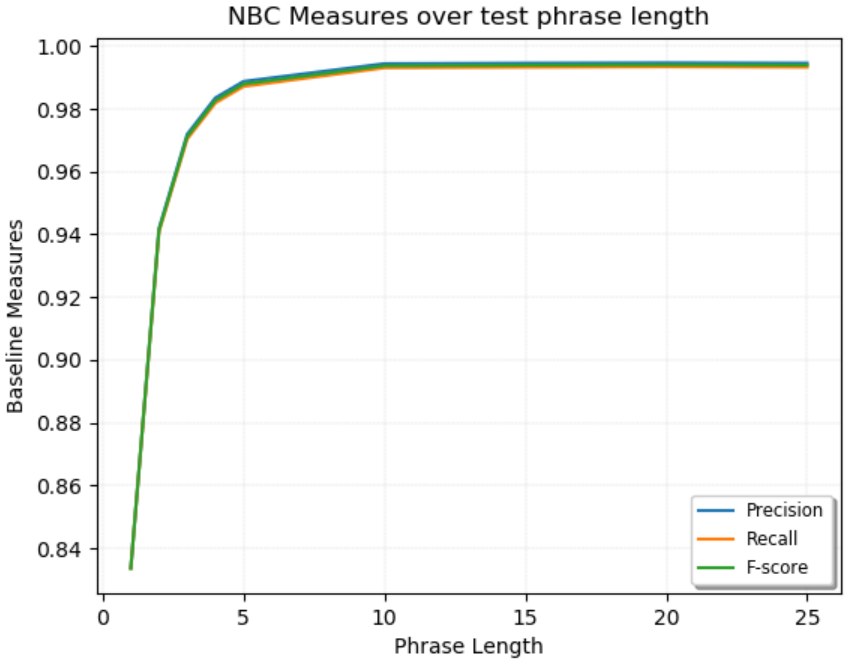

Figure 4.2 NBC matrices over Baseline language model

Based on Figure 4.1 and 4.2 the performance of the classifiers fall below 86% for CFA and 84% for NBC in F-score values respectively for test strings with 1 word length. As the number of words in the test phrase grows from one to two the performance grows exponentially above 94% for both classifiers.

Context II

The Figures 4.3 and 4.4 are the graphical presentations of the relative performance of N-grams measured in F-score.

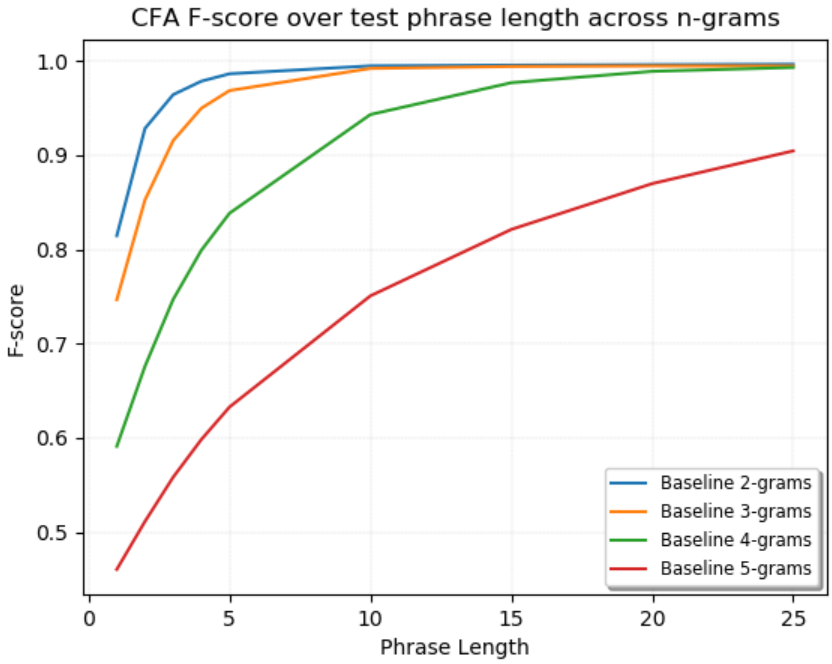

Figure 4.3 The F-score of CFA over Baseline language model across n-grams

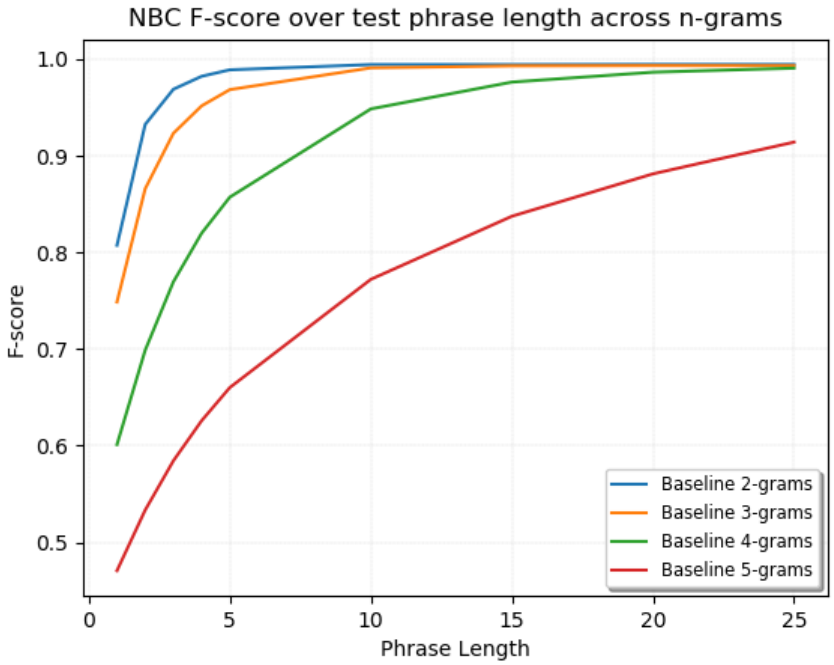

Figure 4.4 The F-score of NBC over Baseline language model across n-grams

From the graphical presentations above in Figure 4.3 and 4.4, the classifiers performance is below 82% and 81% in F-score for CFA and NBC classifiers for 1 word test phrases for 2-grams but for other values of N the classifiers performance fall below 75%.

The BL language model generates CBN from BOW where as depicted in Table 3.1, 20.03% of the corpus consists of words with length less than or equal to 2 characters long. From these words we cannot generate trigrams. But instead we can generate bigrams from word with a length 2 or more characters long. As we move from bigram to trigram less of the words in the corpus will be useful to generate trigrams and hence increases the chance of meeting missing n-grams from the language models during classification to reduce the classifiers performance.

4.5.2 Exp-2 - Fixed Length CBN from Text Strings without Word Tokenization (BY)

In this experiment the language models are developed by using the training text after data cleaning without performing word tokenization.

Context I

The Figures 4.5 and 4.6 are the graphical presentation of the performance metrics for both classifiers.

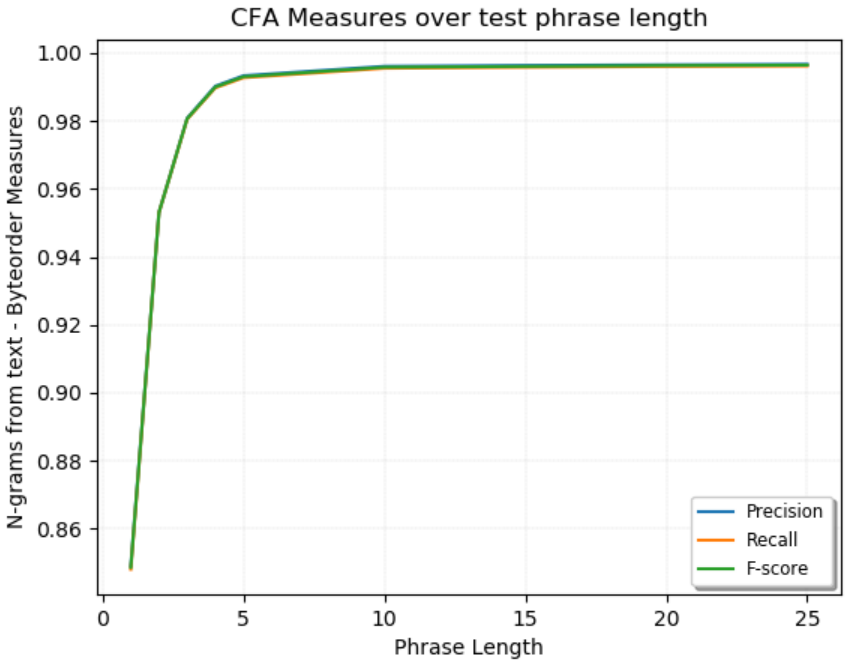

Figure 4.5 CFA over text n-gram language model

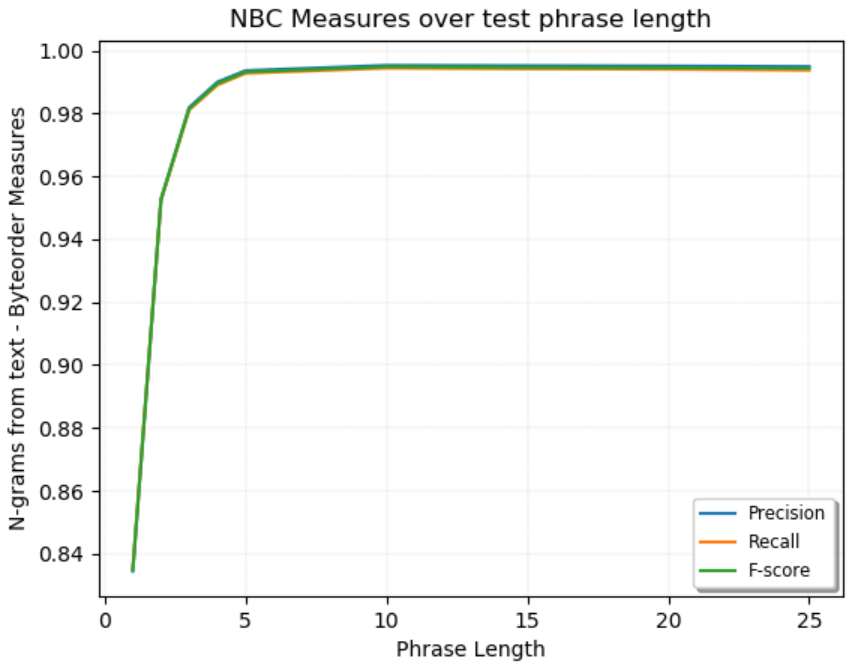

Figure 4.6 NBC metrics over text n-gram language model

Context II

The Figures 4.7 and 4.8 are the graphica presentations on the relative performance of N-grams measured in F-score.

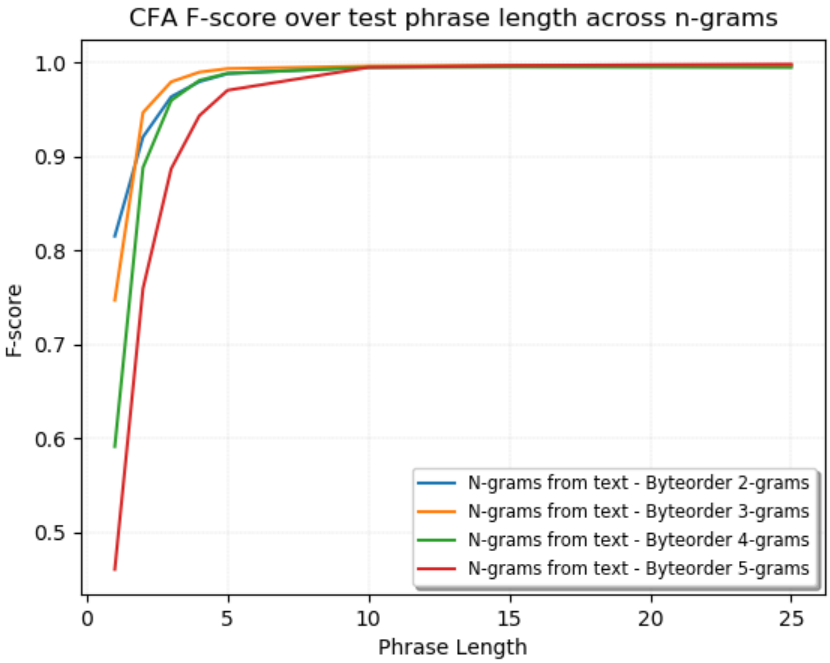

Figure 4.7 The F-score of CFA over text n-grams language model across n-grams

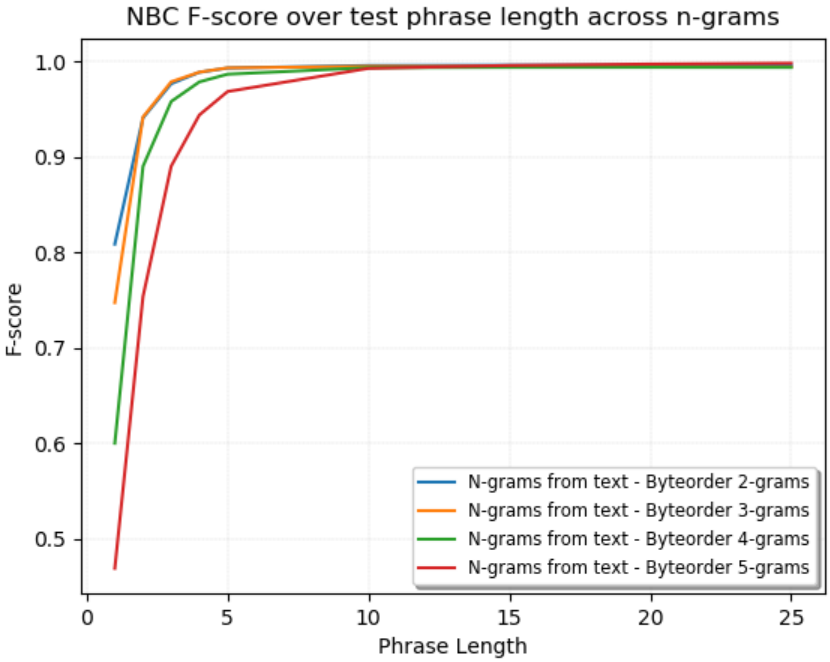

Figure 4.8 The F-score of NBC over text n-grams language model across n-grams

From the graphical presentations above in Figure 4.7 and 4.8, where n=2 is performing relatively better for short and long test strings. For 1 word test string the performance of the classifiers is around 80% which is lower than the BL language model.

From the graphical presentations above in Figure 4.7 and 4.8, the classifiers performance is below 83% for 1 word test phrases for 2-grams. But for other values of N the classifiers performance fall below 75%.

This language modelling technique provides for larger possible n-gram feature extraction as compared to other CBN language models depicted in Table 3.1. It goes beyond word boundaries to generate n-grams including white space as valid character. This language model generates the largest number of features from the same amount of text as compared to other language models we considered in this study.

The probability of a word sequence in a test string similar to the training corpus will be lower and hence the probability of CBN to match during classification. This increases the chance of missing n-grams to reduce the performance of classifiers.

4.5.3 Exp-3 – Fixed Length CBN from BOW with Location Features (FL)

This is the n-gram model generated from the BOW generated after cleaning and tokenization considering the location features.

Unlike the BL language model FL generates CBN from BOW where n-gram is generated from all the smaller test strings. When a 5-gram is generated, all of the words in the language corpus with length less than 5 characters long will also be considered. This helps to capture as many n-gram features and helps to improve classifiers performance by reducing the chances of meeting missing n-grams during classification.

Context I

The Figures 4.9 and 4.10 are the graphical presentations of the performance metrics for both classifiers.

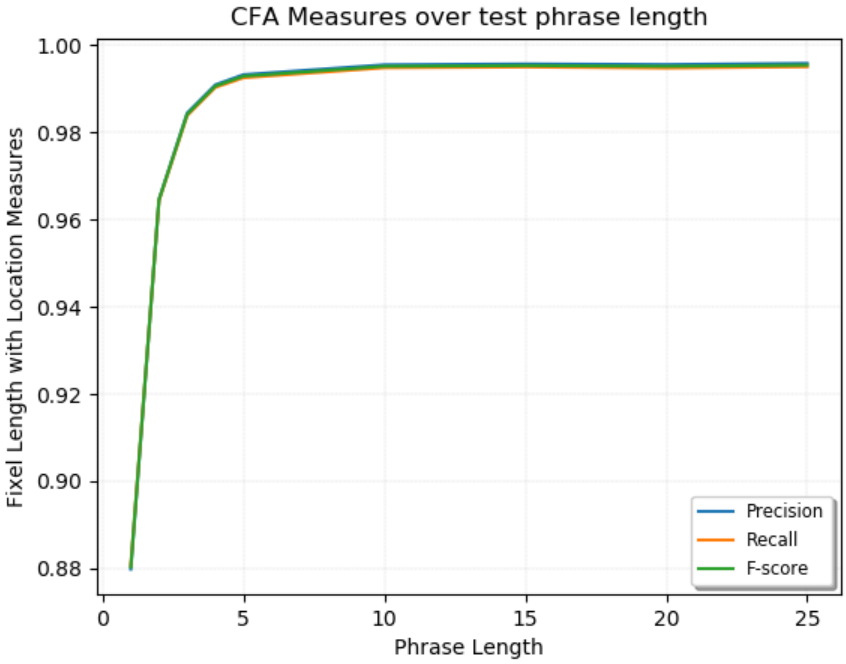

Figure 4.9 CFA over Fixed length n-gram on BOW with location features

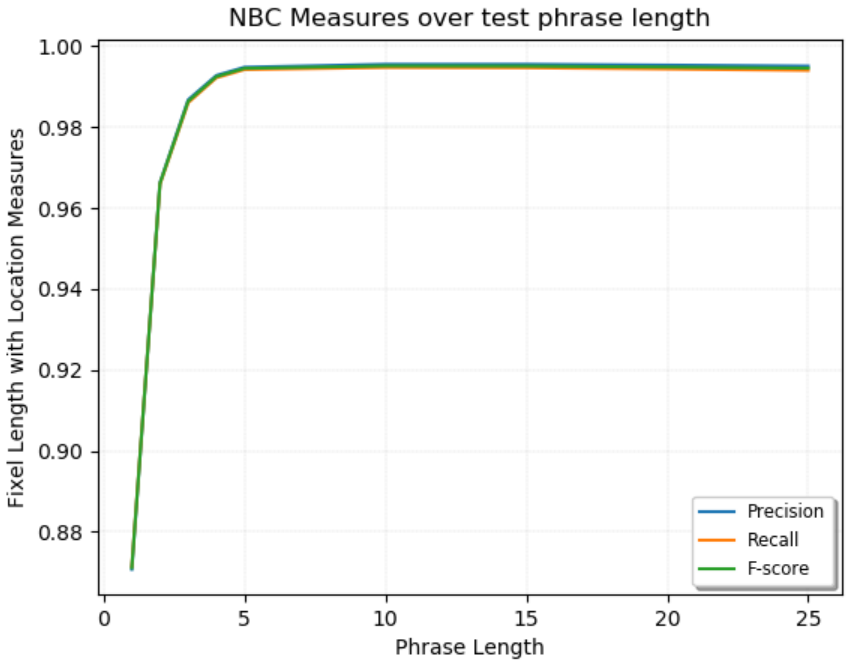

Figure 4.10 NBC over Fixed length n-gram on BOW with location features

In comparison to the language models we have seen in Exp-1 and Exp-2 this classifier performs better in both short and longer test phrases. For CFA the classifier performs above 88% whereas NBC performs above 87% for 1 word test string as depicted in Figure 4.9 and 4.10.

Context II

The Figures 4.11 and 4.12 are the graphica presentations on the relative performance of N-grams measured in F-score.

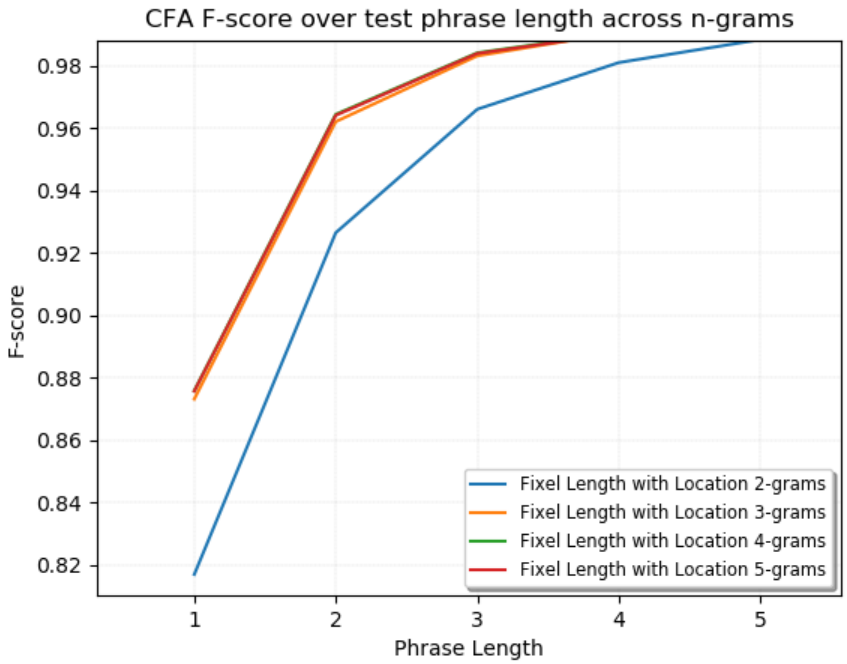

Figure 4.11 F-score of CFA over Fixed length n-gram on BOW with location across n-grams

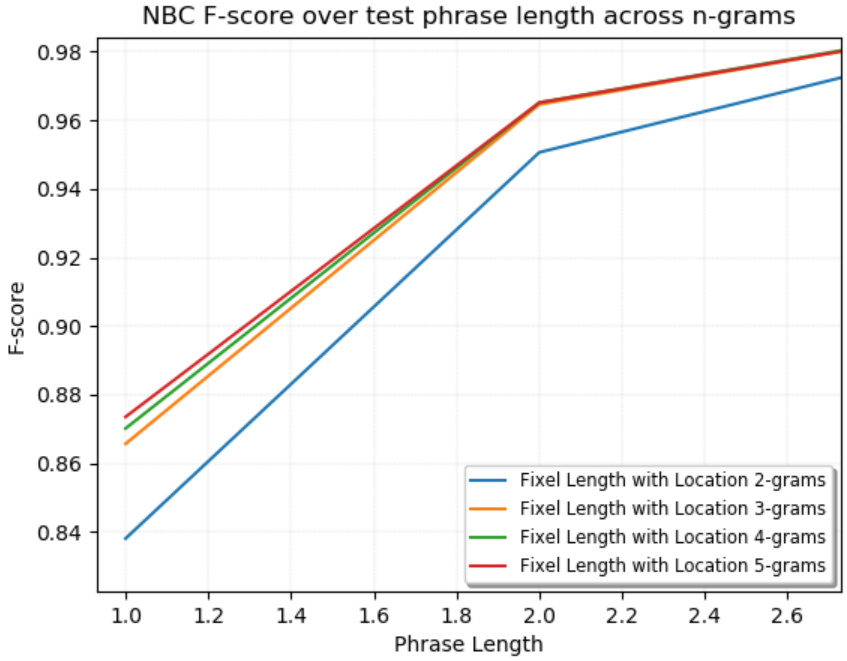

Figure 4.12 F-score of NBC over Fixed length n-gram on BOW with location across n-grams

From the graphical presentations above in Figure 4.11 and 4.12, the classifiers performance is below 82% and 84% in F-score for CFA and NBC classifiers for 1 word test phrases for 2-grams but where N=5 the classifiers performance improved to 88.02% and 87.11% for CFA and NBC relatively. N=5 generates an n-gram form all words shorter than N and hence classifiers perform better due to less chances of meeting missing n-grams during classifications.

4.5.4 Exp-4 – Infinity CBN from of BOW without Location Features (IN)

This is the n-gram model generated from the BOW generated after cleaning and tokenization considering the Infinity n-grams without location features.

Similar to the BL language model, IN generates CBN from BOW where as depicted in Table 3.1, 20.03% of the corpus consists of words with length less than or equal to 2 characters long. From these words we cannot generate trigrams. But instead we can generate bigrams from word with a length 2 or more characters long. As we move from bigram to trigram less of the words in the corpus will be useful to generate trigrams and hence increases the chance of meeting missing n-grams from the language models during classification to reduce the classifiers performance.

Context I

Below are the test results on each performance measures:

The Figures 4.13 and 4.14 are the graphical presentations of the performance metrics for both classifiers.

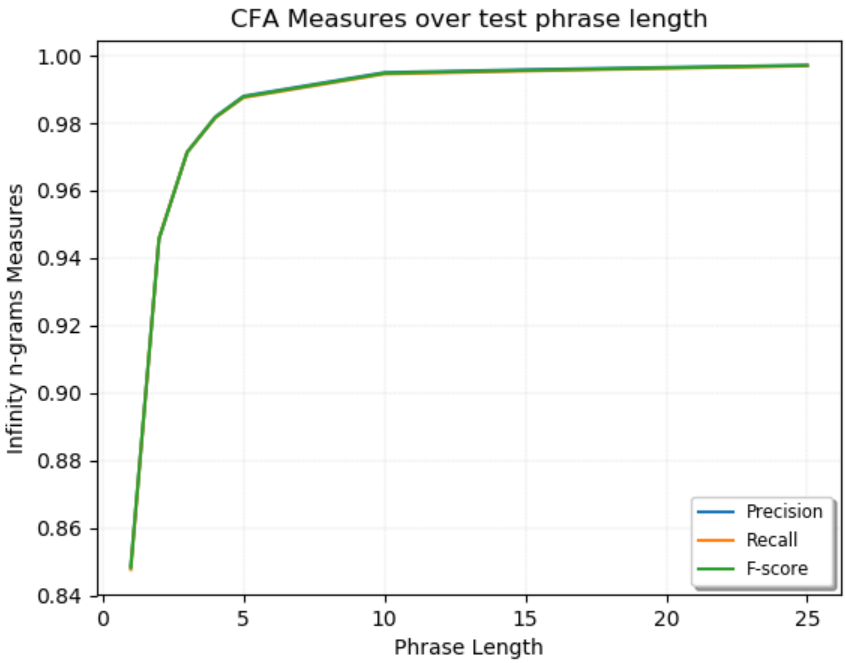

Figure 4.13 CFA over character Infinity n-gram without location features

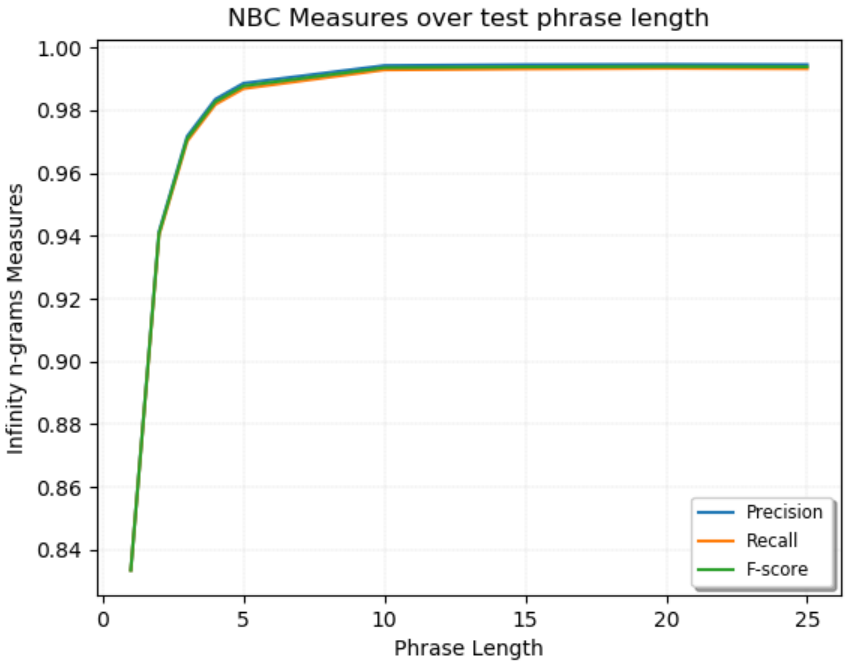

Figure 4.14 NBC over character Infinity n-gram without location features

Based on these figures 4.13 and 4.14 the performance of the classifiers fall below 86% for CFA and 84% for NBC in F-score values respectively for test strings with 1 word length. As this language model does not include shorter words to gnerate higher order n-grams, it performes in

a similar range of the BL language model. As the number of words in the test phrase grows from one to two the performance grows exponentially above 94% for both classifiers.

Context II

The Figures 4.15 and 4.16 are the graphica presentations on the relative performance of N-grams measured in F-score.

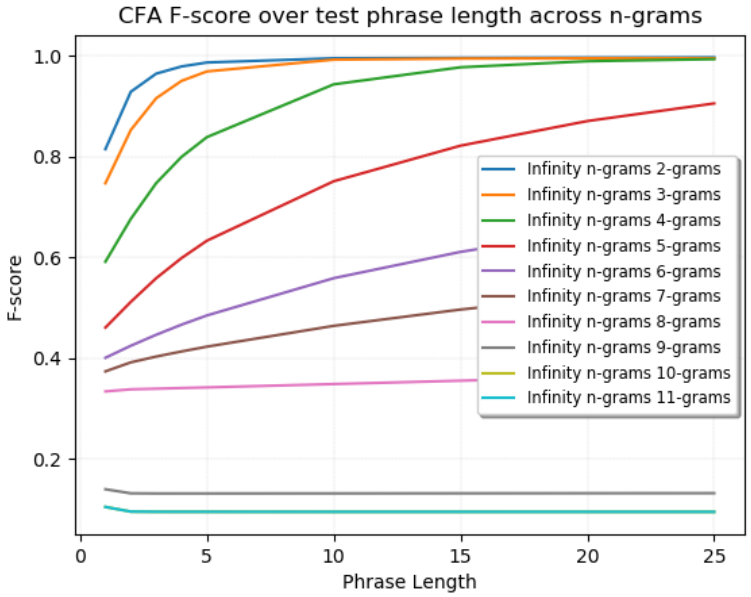

Figure 4.15 F-score of CFA over Infinity n-gram without location features across n-grams

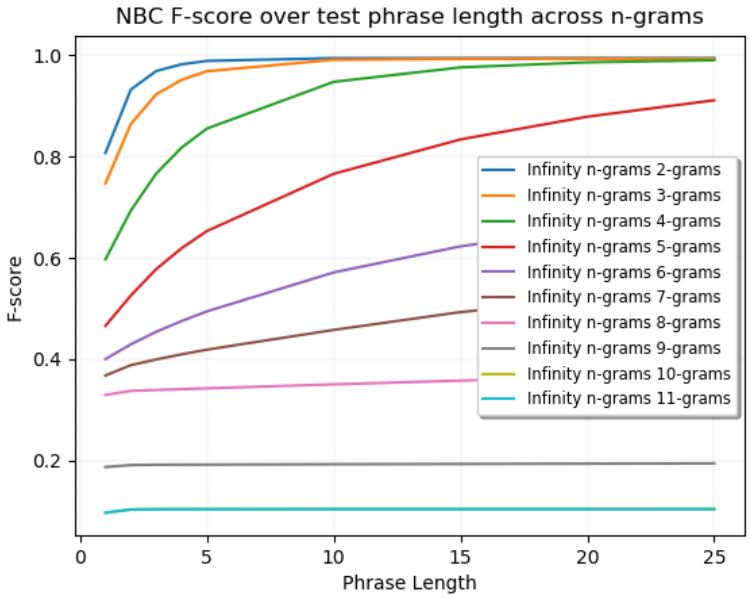

Figure 4.16 F-score of NBC over Infinity n-gram without location features across n-grams

From the graphical presentations above in Figure 4.15 and 4.16, the 2-gram performs above 80% F-score for 1 word test phrase and maintains higher performance for longer test phrases and thus N=2 is performing better than other values of N for similar reasons indicated for the BL language model under experiment Exp-1.

4.5.5 Exp-5 - Infinity CBN from BOW with Location Features (IL)

This is the n-gram model generated from the BOW generated after cleaning and tokenization considering the Infinity n-grams with location features.

Like IN language model generates CBN from BOW where this IL does not generate higher order n-grams for shorter words. Due to the location features, it generates very larger number of n-gram features from training and testing sets.

Context I

Below are the test results on each performance measures:

The Figures 4.17 and 4.18 are the graphical presentations of the performance metrics for both classifiers.

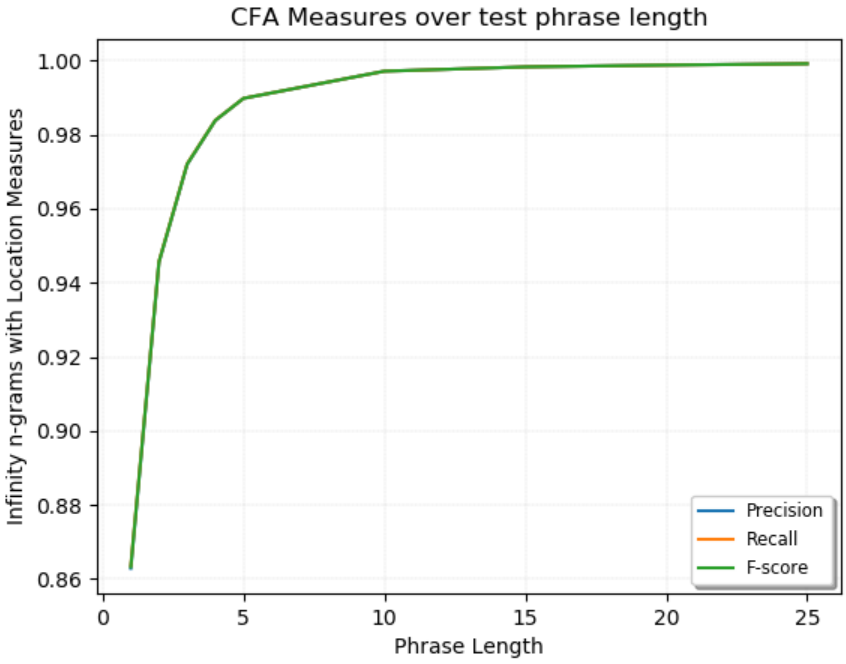

Figure 4.17 CFA matrices over Infinity n-gram with location features language model

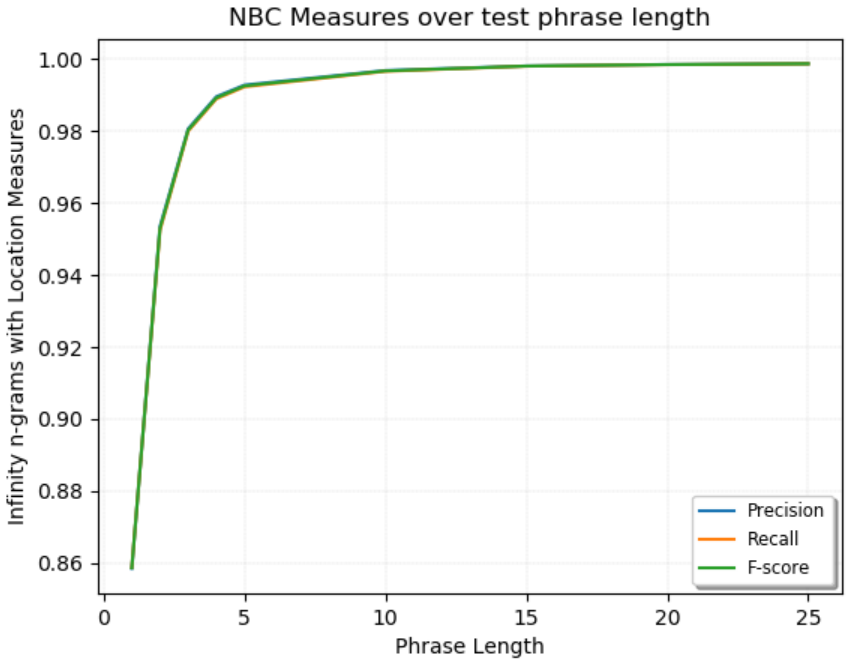

Figure 4.18 NBC matrices over Infinity n-gram with location features language model

Based on these figures 4.17 and 4.18 the performance of the classifiers fall below 87% for CFA and 86% for NBC in F-score values respectively for test strings with 1 word length. Unlike the FL language model it does not include shorter words to gnerate higher order n-grams with location features, thus it performes better next to the FL language model. As the number of words in the test phrase grows from one to two the performance grows exponentially above 94% for both classifiers.

Context II

The Figures 4.19 and 4.20 are the graphica presentations on the relative performance of N-grams measured in F-score.

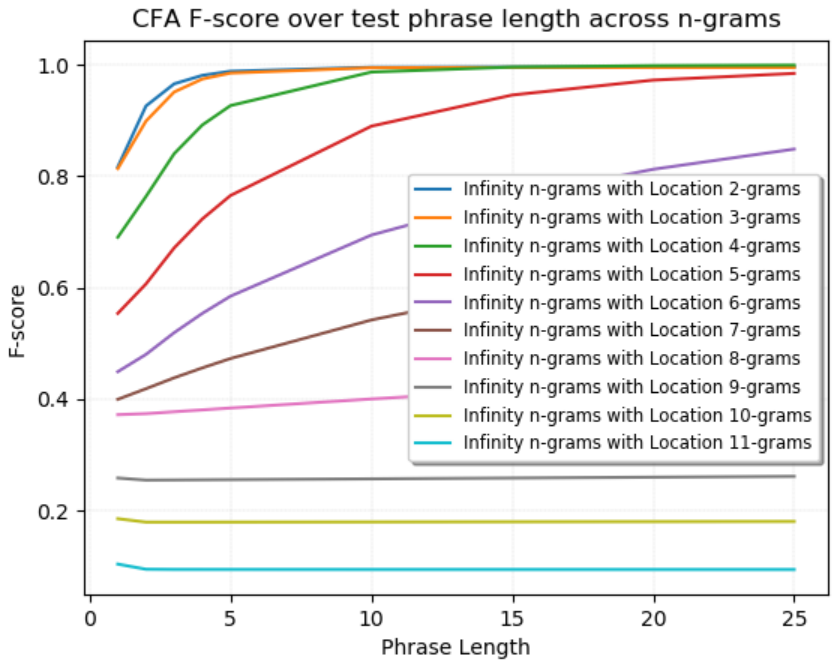

Figure 4.19 CFA matrices over Infinity n-gram with location features over n-grams

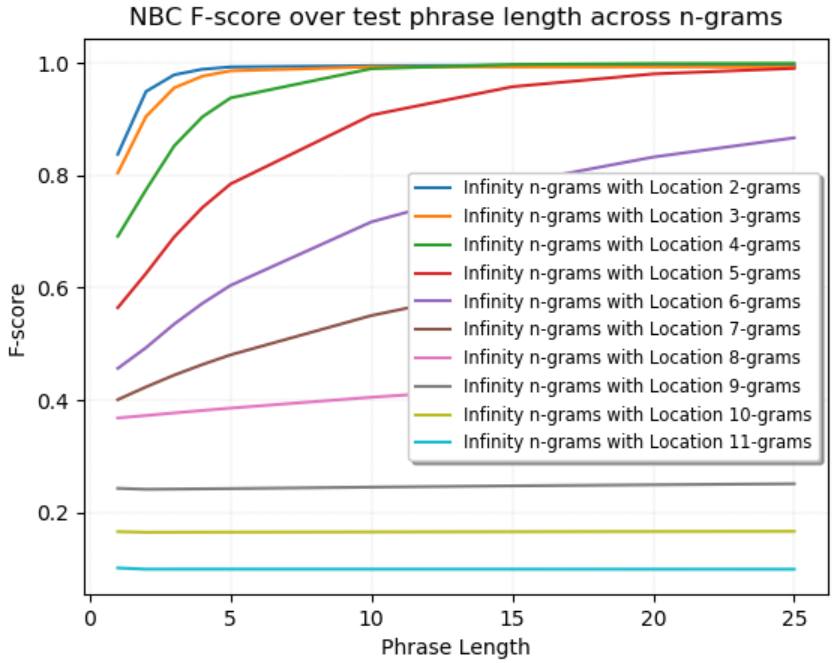

Figure 4.20 NBC matrices over Infinity n-gram with location features over n-grams

From the graphical presentations above in Figure 4.19 and 4.20, the 2-gram performs above 80% F-score for 1 word test phrase and maintains higher performance for longer test phrases and thus N=2 is performing better than other values of N for similar reasons indicated for the FL language model under experiment Exp-3.

4.6 Cross Model Comparisons

Shorter test phrase lengths provides for less n-gram features to extract from them. The probability of these features occurring in all language models is higher. Classifiers depend on only the frequency differences of these few features across the language models to predict one out of the possible languages and thus reduces the classifiers performance. As the phrase length grows, more features provides for better estimates to predict the language of the test document.

The classifiers' performance based on F-score measures across the different testing language models is summarized in figures below:

Context I

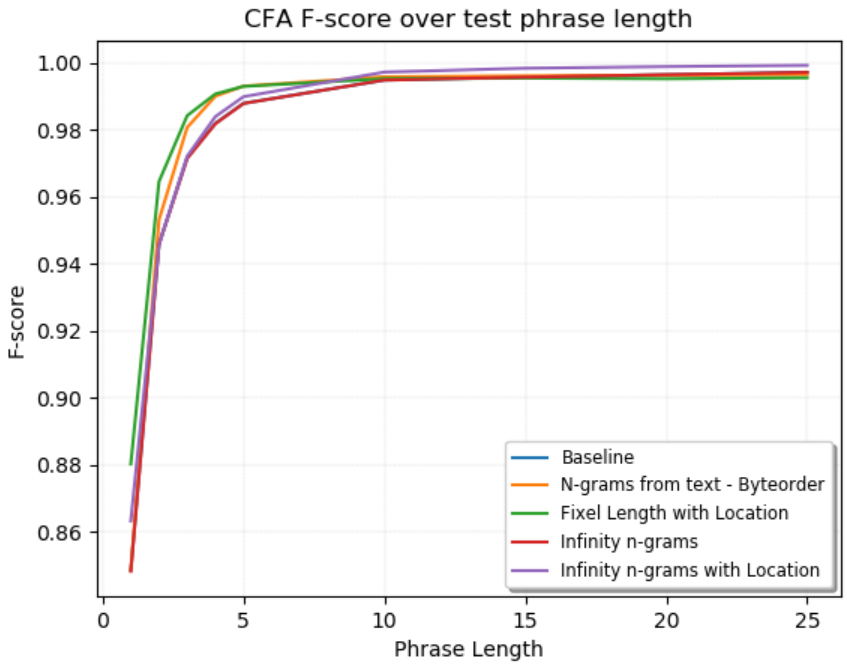

Figure 4.21 F-score of CFA over experimenting language models

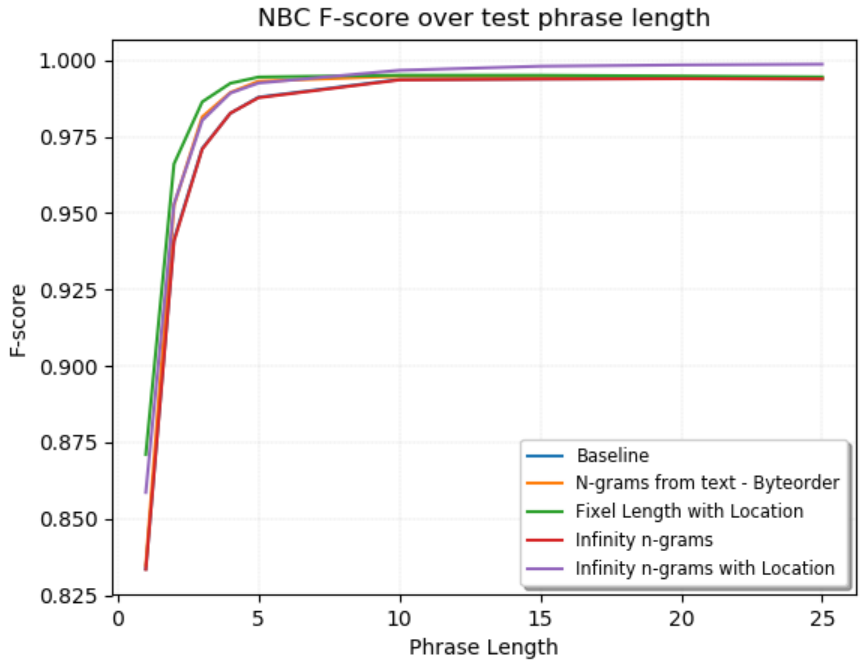

Figure 4.22 F-score of NBC over experimenting language models

Based on these figures 4.21 and 4.22 the performance of the FL language model is above 88% for CFA and 87% for NBC in F-score values respectively for test strings with 1 word length as compared to these classfiers performance in other language models.

Context II

CFA Classifier

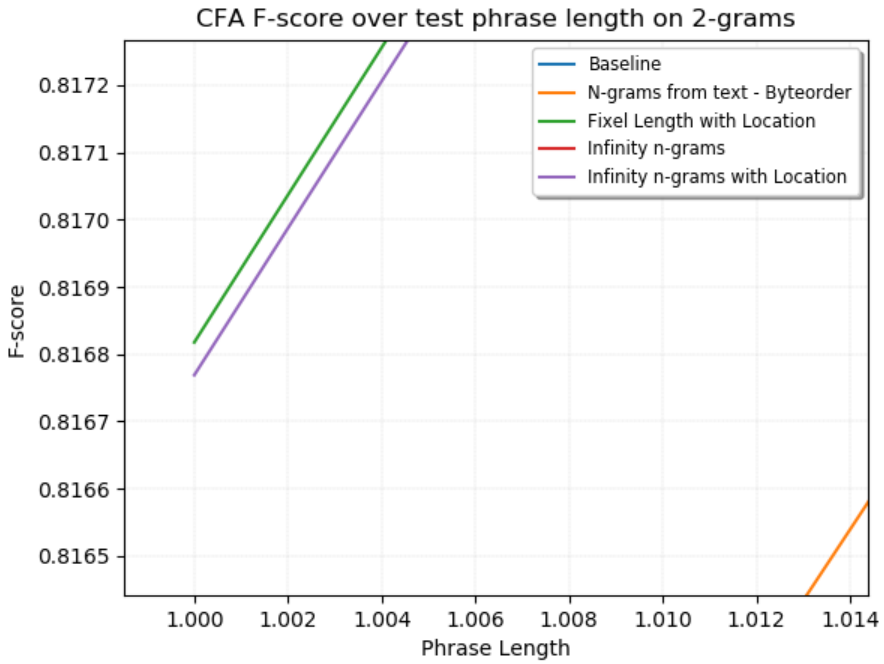

Figure 4.23 F-score of CFA over experimenting language models on 2-grams

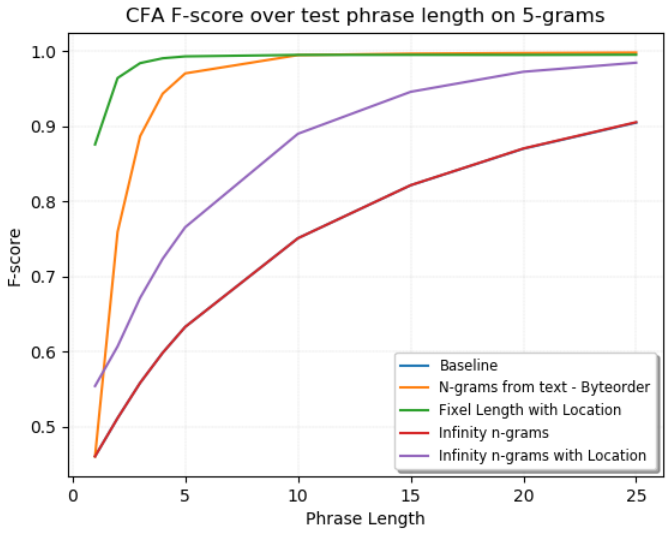

Figure 4.24 F-score of CFA over experimenting language models on 5-grams

Based on these figures 4.23 and 4.24 the performance of the FL language model is above 87% for CFA in F-score values for test strings with 1 word length and N=5 as compared to the classfiers performance when N=2 where it performs below 82% on the same test phrase lenght.

NBC Classifier

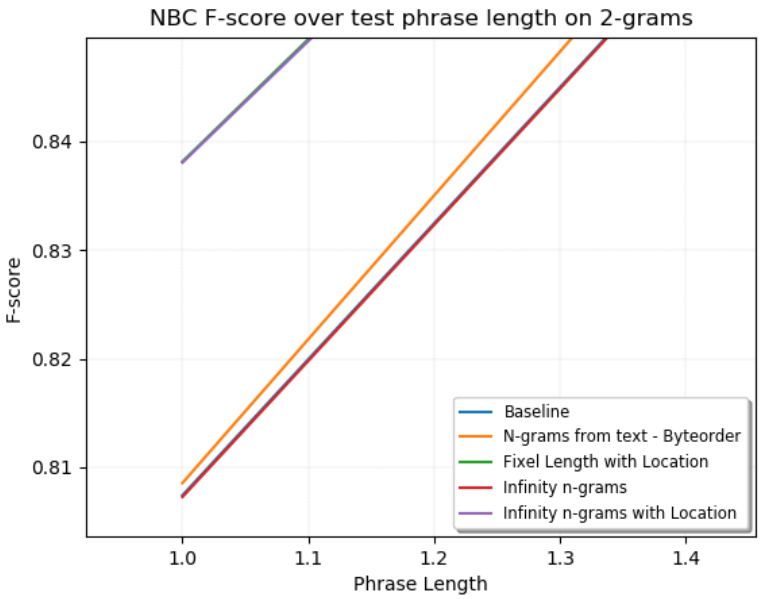

Figure 4.25 F-score of NBC over experimenting language models on 2-grams

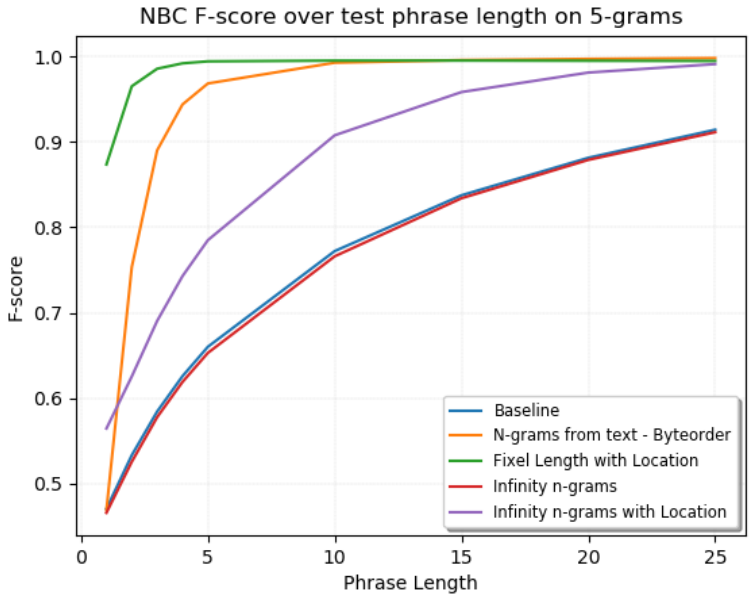

Figure 4.26 F-score of NBC over experimenting language models on 5-grams

Based on these figures 4.25 and 4.26 the performance of the FL language model is above 87% for NBC in F-score values for test strings with 1 word length and N=5 as compared to the classfiers performance when N=2 where it performs below 84% on the same test phrase lenght.

4.7 Summary

Context I

	Fixed length n-rams without location feature language model (BL)			Text n-grams features based language model (BY)			Fixed length n-rams with location features language model (FL)			Infinity n-grams with location feature language model (IL)			Infinity n-grams without location feature language model (IN)		
Words	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
1	84.90%	84.79%	84.85%	84.92%	84.81%	84.86%	87.98%	88.07%	88.02%	86.28%	86.36%	86.32%	84.88%	84.77%	84.82%
2	94.58%	94.60%	94.59%	95.31%	95.31%	95.31%	96.47%	96.44%	96.45%	94.56%	94.58%	94.57%	94.57%	94.58%	94.57%
3	97.16%	97.15%	97.16%	98.08%	98.05%	98.07%	98.45%	98.39%	98.42%	97.20%	97.22%	97.21%	97.14%	97.14%	97.14%
4	98.20%	98.17%	98.19%	99.02%	98.97%	99.00%	99.09%	99.03%	99.06%	98.39%	98.39%	98.39%	98.19%	98.16%	98.17%
5	98.81%	98.76%	98.79%	99.33%	99.27%	99.30%	99.33%	99.26%	99.29%	98.98%	98.98%	98.98%	98.80%	98.76%	98.78%
10	99.50%	99.46%	99.48%	99.61%	99.55%	99.58%	99.55%	99.48%	99.52%	99.72%	99.72%	99.72%	99.50%	99.46%	99.48%
15	99.59%	99.55%	99.57%	99.63%	99.58%	99.60%	99.57%	99.50%	99.54%	99.83%	99.83%	99.83%	99.59%	99.55%	99.57%
20	99.65%	99.62%	99.64%	99.65%	99.60%	99.63%	99.56%	99.48%	99.52%	99.88%	99.88%	99.88%	99.65%	99.62%	99.64%
25	99.72%	99.70%	99.71%	99.67%	99.62%	99.64%	99.59%	99.51%	99.55%	99.92%	99.92%	99.92%	99.72%	99.70%	99.71%
Avg.	**96.90%**	**96.87%**	**96.88%**	**97.25%**	**97.19%**	**97.22%**	**97.73%**	**97.68%**	**97.71%**	**97.20%**	**97.21%**	**97.20%**	**96.89%**	**96.86%**	**96.88%**

Table 4.4 CFA performance over language models and phase lengths

The F-score for test phrases length of 1 word reached the maximum of 88.02% for the CFA classifier, considering all n-grams n=2, 3, 4 and 5 on Fixed length n-grams with location features language model.

The classifier improves its performance as measured by its F-score, as the test phrase length grows to 2 words its performance reaches to 96.45% as highlighted in green on Table 4.4.

	Fixed length n-rams without location feature language model (BL)			Text n-grams features based language model (BY)			Fixed length n-rams with location features language model (FL)			Infinity n-grams with location feature language model (IL)			Infinity n-grams without location feature language model (IN)		
Words	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
1	83.36%	83.36%	83.36%	83.45%	83.53%	83.49%	87.07%	87.14%	87.11%	85.85%	85.88%	85.87%	83.35%	83.35%	83.35%
2	94.19%	94.08%	94.14%	95.27%	95.24%	95.26%	96.63%	96.59%	96.61%	95.35%	95.25%	95.30%	94.13%	94.03%	94.08%
3	97.19%	97.04%	97.11%	98.19%	98.11%	98.15%	98.68%	98.60%	98.64%	98.07%	98.00%	98.03%	97.17%	97.02%	97.10%
4	98.35%	98.19%	98.27%	99.00%	98.90%	98.95%	99.28%	99.22%	99.25%	98.96%	98.90%	98.93%	98.35%	98.19%	98.27%
5	98.88%	98.71%	98.80%	99.36%	99.27%	99.32%	99.48%	99.42%	99.45%	99.28%	99.23%	99.26%	98.86%	98.69%	98.78%
10	99.44%	99.31%	99.38%	99.53%	99.44%	99.48%	99.56%	99.47%	99.51%	99.68%	99.66%	99.67%	99.43%	99.29%	99.36%
15	99.46%	99.33%	99.39%	99.52%	99.41%	99.47%	99.56%	99.47%	99.51%	99.81%	99.81%	99.81%	99.46%	99.32%	99.39%
20	99.47%	99.34%	99.41%	99.51%	99.40%	99.45%	99.53%	99.44%	99.49%	99.85%	99.85%	99.85%	99.47%	99.34%	99.40%
25	99.46%	99.33%	99.39%	99.49%	99.37%	99.43%	99.51%	99.40%	99.46%	99.87%	99.87%	99.87%	99.46%	99.33%	99.39%
Avg.	**96.64%**	**96.52%**	**96.58%**	**97.03%**	**96.96%**	**97.00%**	**97.70%**	**97.64%**	**97.67%**	**97.41%**	**97.38%**	**97.40%**	**96.63%**	**96.51%**	**96.57%**

Table 4.5 NBC performance over language models and phase lengths

The F-score for test phrases length of 1 word reached the maximum of 87.11% for the NBC classifier, considering all n-grams n=2, 3, 4 and 5 on Fixed length n-grams with location features language model.

The classifier improves its performance as measured by its F-score, as the test phrase length grows to 2 words its performance reaches to 96.61% as highlighted in green on Table 4.5.

Context II

	Fixed length n-rams without location feature 2-grams (BL)			Text n-grams features based language model 2-grams (BY)			Fixed length n-rams with location features language model 5-grams (FL)			Infinity n-grams with location feature language model 2-grams (IL)			Infinity n-grams without location feature language model 2-grams (IN)		
Words	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
1	81.56%	81.37%	81.46%	81.60%	81.41%	81.51%	87.53%	87.59%	87.56%	81.65%	81.71%	81.68%	81.56%	81.37%	81.46%
2	92.84%	92.86%	92.85%	92.07%	92.05%	92.06%	96.43%	96.40%	96.41%	92.63%	92.59%	92.61%	92.83%	92.85%	92.84%
3	96.44%	96.45%	96.44%	96.35%	96.33%	96.34%	98.43%	98.36%	98.40%	96.60%	96.57%	96.58%	96.44%	96.45%	96.44%
4	97.87%	97.87%	97.87%	97.96%	97.94%	97.95%	99.08%	99.01%	99.04%	98.11%	98.09%	98.10%	97.87%	97.87%	97.87%
5	98.66%	98.65%	98.66%	98.80%	98.79%	98.80%	99.33%	99.26%	99.29%	98.85%	98.83%	98.84%	98.66%	98.65%	98.66%
10	99.51%	99.49%	99.50%	99.56%	99.53%	99.55%	99.53%	99.45%	99.49%	99.57%	99.54%	99.56%	99.51%	99.49%	99.50%
15	99.58%	99.57%	99.58%	99.62%	99.59%	99.61%	99.54%	99.46%	99.50%	99.62%	99.59%	99.60%	99.58%	99.57%	99.58%
20	99.64%	99.62%	99.63%	99.67%	99.64%	99.65%	99.53%	99.44%	99.49%	99.66%	99.64%	99.65%	99.64%	99.62%	99.63%
25	99.68%	99.66%	99.67%	99.66%	99.62%	99.64%	99.56%	99.47%	99.52%	99.65%	99.62%	99.64%	99.68%	99.66%	99.67%
Avg.	**96.20%**	**96.17%**	**96.19%**	**96.14%**	**96.10%**	**96.12%**	**97.66%**	**97.60%**	**97.63%**	**96.26%**	**96.24%**	**96.25%**	**96.20%**	**96.17%**	**96.18%**

Table 4.6 CFA performance over language models

Fixed length n-rams without location feature 2-grams indicated as (BL) and Infinity n-grams without location feature language model 2-grams indicated as (IN) language models provides for identical performance as the same n-gram is considered on both cases under this context.

From all language models, as highlighted in green on Table 4.6, the F-score for test phrases length of 1 word reached the maximum of 87.56% for the CFA classifier, considering all 5-grams on fixed length n-grams with location features language model. The classifier improves its performance as the test phrase length grows to 2 words its performance reaches 96.41%.

Of all the language models used in this study, N=5 on Fixed length n-rams with location features language model is the most optimal whereas N=2 gives the optimal value of N for the remaining language models using the CFA classification method.

	Fixed length n-rams without location feature 2-grams (BL)			Text n-grams features based language model 2-grams (BY)			Fixed length n-rams with location features language model 5-grams (FL)			Infinity n-grams with location feature language model 2-grams (IL)			Infinity n-grams without location feature language model 2-grams (IN)		
Words	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
1	80.75%	80.74%	80.74%	80.83%	80.89%	80.86%	87.33%	87.38%	87.36%	83.79%	83.82%	83.80%	80.74%	80.71%	80.73%
2	93.27%	93.24%	93.25%	94.09%	94.07%	94.08%	96.54%	96.49%	96.51%	95.04%	94.92%	94.98%	93.25%	93.23%	93.24%
3	96.93%	96.87%	96.90%	97.67%	97.64%	97.65%	98.61%	98.52%	98.56%	98.00%	97.91%	97.96%	96.93%	96.87%	96.90%
4	98.28%	98.19%	98.24%	98.88%	98.84%	98.86%	99.23%	99.16%	99.19%	98.98%	98.92%	98.95%	98.28%	98.19%	98.24%
5	98.95%	98.86%	98.91%	99.32%	99.28%	99.30%	99.46%	99.39%	99.43%	99.38%	99.32%	99.35%	98.95%	98.86%	98.91%
10	99.49%	99.41%	99.45%	99.60%	99.55%	99.57%	99.55%	99.47%	99.51%	99.61%	99.56%	99.58%	99.49%	99.41%	99.45%
15	99.50%	99.40%	99.45%	99.62%	99.57%	99.60%	99.56%	99.47%	99.52%	99.66%	99.62%	99.64%	99.50%	99.40%	99.45%
20	99.52%	99.42%	99.47%	99.62%	99.57%	99.60%	99.54%	99.45%	99.49%	99.64%	99.59%	99.62%	99.51%	99.42%	99.46%
25	99.51%	99.41%	99.46%	99.63%	99.58%	99.60%	99.53%	99.42%	99.47%	99.62%	99.57%	99.60%	99.50%	99.40%	99.45%
Avg.	**96.24%**	**96.17%**	**96.21%**	**96.58%**	**96.55%**	**96.57%**	**97.71%**	**97.64%**	**97.67%**	**97.08%**	**97.03%**	**97.05%**	**96.24%**	**96.17%**	**96.20%**

Table 4.7 NBC performance over language models

Fixed length n-rams without location feature 2-grams indicated as (BL) and Infinity n-grams without location feature language model 2-grams indicated as (IN) language models provides for identical performance as the same n-gram is considered on both cases under this context.

As highlighted in green on Table 4.7, the F-score for test phrases length of 1 word reached the maximum of 87.36% for that NBC classifier, considering all 5-grams on fixed length n-grams with location features language model. The classifier improves its performance as measured in F-score, as the test phrase length grows to 2 words its performance reaches 96.51%.

Of all the language models used in this study, N=5 on Fixed length n-rams with location features language model is the most optimal whereas N=2 gives the optimal value of N for the remaining language models using the NBC classification method.

The detail compressions of the two classifiers using F-score under each evaluation context across the test phrase length for the highest performing language model (FL) and the optimal N-gram is detailed in Table 4.8 below.

	Context I			Context II		
Words	**CFA**	**NBC**	**CFA - NBC**	**CFA**	**NBC**	**CFA - NBC**
1	88.02%	87.11%	0.91%	87.56%	87.36%	0.20%
2	96.45%	96.61%	-0.16%	96.41%	96.51%	-0.10%
3	98.42%	98.64%	-0.22%	98.40%	98.56%	-0.16%
4	99.06%	99.25%	-0.19%	99.04%	99.19%	-0.15%
5	99.29%	99.45%	-0.16%	99.29%	99.43%	-0.14%
10	99.52%	99.51%	0.01%	99.49%	99.51%	-0.02%
15	99.54%	99.51%	0.03%	99.50%	99.52%	-0.02%
20	99.52%	99.49%	0.03%	99.49%	99.49%	0.00%
25	99.55%	99.46%	0.09%	99.52%	99.47%	0.05%
Average	**97.71%**	**97.67%**	**0.04%**	**97.63%**	**97.67%**	**-0.04%**

Table 4.8 Detailed comparison of classifiers across evaluation context on (FL)

The comparative performance of classifiers over the evaluation contexts for the highest performing language model, i.e. Fixed length character n-gram with location features (FL), considering N=5 under Context II highlighted in green on Table 4.8 is summarized on Table 4.9.

	Context I (I)		Context II (II)		I – II	
Classifiers	**Short**[13]	**Overall**[14]	**Short**	**Overall**	**Short**	**Overall**
CFA	88.02%	97.71%	87.56%	97.63%	0.46%	0.08%
NBC	87.11%	97.63%	87.36%	97.67%	-0.25%	-0.04%
CFA-NBC	0.91%	0.08%	0.20%	-0.04%		

Table 4.9 Comparative performance of classifiers over the best performing model (FL)

Based on Table 4.9, comparing the performance a classifier across the two contexts, CFA performs better in Context I with higher percentage points differences on short texts, whereas NBC performs better in Context II.

Based on Table 4.9, comparing the highest performance values highlighted in green against the non-highlighted row entries, in three out of the 4 comparisons, CFA performs better than NBC.

[13] Short refers to a 1 word test phrase length
[14] Overall refers to the average performance of the classifier over the test phrase length of 1, 2, 3, 4, 5, 10, 15, 20, and 25

We have tested both classifiers using the Fixed Length CBN language model with test phrases generated from news sites[15] for Amharic and Tigrigna, and text extracted from different books of the bible for Geez and Guragigna not used on the training sets. An average of 7,000 test phrases were generated from 1Kb text after data cleaning for each languages for testing.

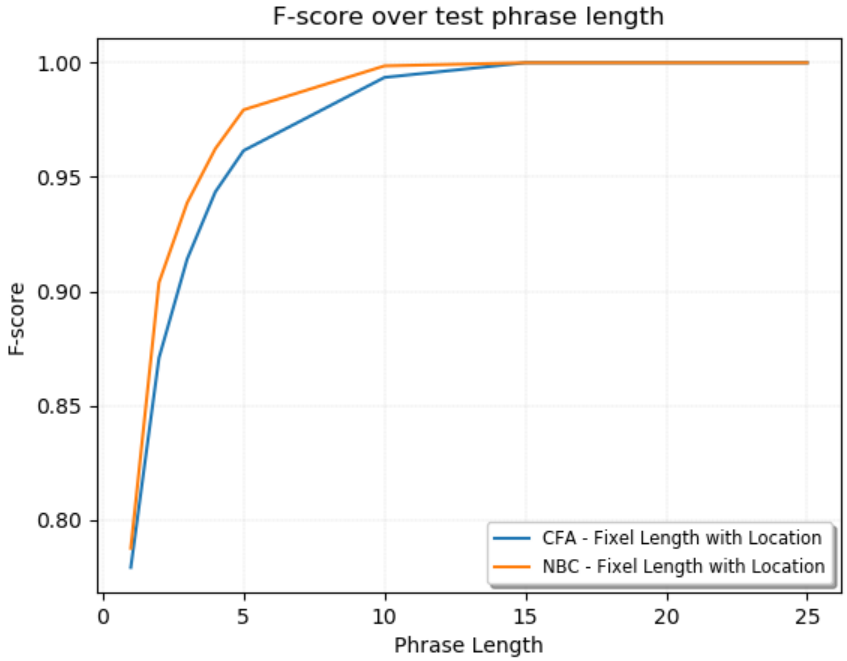

Figure 4.27 Comparative performance of CFA and NBC classifiers

The classifiers performance in F-score reduced to 77.91% and 78.74% for CFA and NBC respectively on both evaluation contexts (N=5 on context II) on unseen test phrases extracted from different source. NBC performs relatively better from CFA on this scenario.

[15] The Amharic news articles are collected from https://www.waltainfo.com/ and the Tigrigna news articles are collected from https://tigrigna.voanews.com/

CHAPTER FIVE

CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusions

The dominant language groups under the family of Ethio-Semitic languages are Amharic, Geez, Guragigna and Tigrigna. The wider use of advanced ICT brought a huge advantage to easily produce and communicate documents electronically, empowered language users to generate more documents in their native languages. As more and more documents are produced and exchanged between language boundaries, there comes a need for the system of exchanges to become language agnostic to serve wider audiences from diverse culture and language background. With this comes the need for systems and applications to automatically detect the language of documents before performing further processing specific to the language of the document at hand.

In this research we have performed a comparative study of CFA and NBC classifiers over a training and test setting by making changes to the language models. We have studied 5 types of language models, specifically, Fixed length CBN (n=2, 3, 4 and 5) with and without location features applied on a BOW, Infinity n-grams (n starts from 2) with and without location features applied on a BOW, and also Fixed length CBN (n=2, 3, 4 and 5) applied on source text without word tokenization language models.

We have used 10-fold cross validation to compute performance measures (Precision, Recall, F-score) from the averages of the 10 round tests performed for each classifiers under each evaluation context and language models.

The experiments are conducted on two different context. One is considering all the features with n= (2, 3, 4 and 5) during the classification process, referred in this study as Context I. The second is, the commonly used approach of identifying the most optimal N value for classifiers by testing the performance of classifiers on each values of n, where n= (2, 3, 4 and 5), which we referred it as Context II.

In both evaluation context, our classifiers commonly exhibited higher performance as measured in F-score when the length of the test phrase grows from a single word to 96.61% for 2 words and beyond 99% for 4 words test phrase lengths.

As summarized on tables 4.4 to 4.7 both CFA and NBC classifiers performed very close under each evaluation context corresponding to the language models, phrase length and n-grams features used during classifications.

On comparative tests made on European languages summarized in Figure 2.3 both CFA and NBC classifiers performed 97.59 and 88.66% accuracy respectively for test strings of 50 characters long. The similar test we have made on Ethio-Semitic languages reached above 98% in F-score on test phase length of 3 words which is below an average of 13 UTF-8 characters long. These classifies performed much better on Ethio-Semitic languages as compared to their performance on European languages.

The language model, fixed length n-rams with location features (FL), exhibited higher performance measured in F-score for both classifiers under each evaluation context.

Under Context II, N=5 on Fixed length n-rams with location features language model is the optimal value whereas N=2 is the optimal value of N for the remaining language models for both classifiers.

Considering the highest performing language model and n-gram, CFA classifier performs better under Context I with an F-score of 88.02% on 1 word test phrase length as compared to and F-score of 87.56% on Context II for the same phrase length. The average F-score of CFA across the different test phrase lengths reached 97.71% and 97.63% under Context I and II respectively. CFA exhibits relatively better performance under Context I both on short text and across different test phrase lengths as compared to its performance on Context II.

Considering the highest performing language model and n-gram, NBC classifier exhibited a maximum F-score of 87.11% on 1 word test phrase length as compared to and F-score of 87.36% on Context II for the same phrase length. The average F-score of NBC across the different test phrase lengths is 97.67% under each evaluation context. NBC exhibits relatively better performance under Context II on short text and across different test phrase lengths as compared to its performance under Context I.

On Context I, using fixed length n-rams with location features language model (FL), CFA performed an F-score of 88.02% above NBC which scored 87.11% for 1 word length test strings. The average F-score across the different length test strings 97.71% and 97.67 for CFA and NBC respectively.

On Context II, from all language models, the F-score for test phrases length of 1 word reached the maximum of 87.56% for the CFA classifier, considering all 5-grams on fixed length character n-grams with location features language model. The classifier improves its performance as measured by its F-score to reach a total average of 97.63% and above. Comparing this to the performance of NBC classifier the F-score for test phrases length of 1 word reached the maximum of 87.36% considering all 5-grams for the same language model and its performance reached a total average of 97.67%.

Comparatively speaking, both classifiers performs in a close range on short and long test strings corresponding to the evaluation context, language model, and n-grams that the difference in performance measures is less than 1% to be statistically significant to allow us make a choice of one classifier over the other.

CFA is built on a valid theoretical assumption as compared to NBC, that NBC assumes the probabilities of unique N-grams are independent of one other, and this is not a good assumption to begin with for language modelling. Unlike NBC, CFA does not require smoothing for missing n-grams and it also does not require the computation of n-gram frequencies in advance to build the language models. This makes the process of generating language models and classification relatively simpler. Thus, CFA is a better classifier both theoretically and practically as compared to NBC on both evaluation contexts.

5.2 Contribution

Some of the main contributions of this study are the following:

- The five types of language models are tested on their effectiveness to power CFA and NBC classification algorithms which to our knowledge never been tested on Ethio-Semitic languages. From this study it is known that both CFA and NBC performs comparatively on Ethio-Semitic languages across the different language models considered.

- In European language studies CFA performance better than NBC for languages written in Non-Latin characters. From this study it is known that, the performance of CFA is similar with NBC on Ethiopic, which is a Non-Latin writing system.

- From this study it is known that, the performance of the fixed length character n-grams with location features performs better on both CFA and NBC and classifiers. This knowledge benefits for practical implementations of language identification and Information retrieval systems for better performance.

- Studied the results of a relatively new approach of language modelling based on infinity n-gram and using all n-grams of a given list of windows (n=2, 3, 4 and 5) for classifications. From this study it is known that the relative performance of these modelling and classification methods as compared to the widely used and tested modelling and classification methods.

- Studied the power of language models and classifiers' performance on word level and phrase levels which to our knowledge is not studied with such detail in international researches in general and in Ethio-Semitic languages in particular.

- This study further contributes to the existing knowledge in the areas of language identification and information retrieval on Ethio-Semitic languages towards making them more accessibility on different areas where NLP application are a common place.

5.3 Recommendations

In this investigation we compared the performance of CFA and NBC classifiers across different language models, two different contexts and measure their performances focusing more on their classification power on shorter texts between 1 and 5 words using the classification algorithms performance measures Precision, Recall and F-score. We focused more on F-score as the optimal performance measure as it considers both Precession and Recall. Our test is done on limited aspects of the classifiers and thus we recommend for further studies to be carried out on the following areas:

- Speed is an important element on language identification and retrieval systems. Studying the memory and time complexities of each language models and classifiers will help to see their comparative performance to make this study complete by bringing these knowledge.

- In this investigation we studied Character based fixed length n-grams (n=2, 3, 4 and 5), Infinity n-grams (n starts from 2) with and without location features and text n-grams (n=2, 3, 4 and 5) we recommend further studies of other language modelling techniques such as Graph Based Approach for Language Identification (LIGA).

- In this investigation, we have compared CFA and NBC classifiers. We recommended that comparative studies across different classifiers like, Artificial Neural Network (ANN), Vector Space Model (VSM), Markov Chains, Monet Carlo sampling, Relative Entropy and Ad-Hoc Ranking on different language models.

- Due to the limitation of our development machine we have used to develop and run the project code, a reasonable corpus size is used for training and testing. On the test sets collected from different domain NBC exhibited a relatively better performance then CFA. We recommend to do similar tests on larger dataset collected from different domains to validate the research results.

- Our experiment indicated that Fixed length character n-grams with location features performs better in identifying the language of shorter texts as compared to other models on both classifiers. We recommend further studies to be done across different classifiers to validate and recommend this language model for common use.

REFERENCES

1. Thomas Gottron and Nedim Lipka, 2010, A Comparison of Language Identification Approaches on Short, Query-Style Texts, Proceeding ECIR'2010 Proceedings of the 32ndEuropean conference on Advances in Information Retrieval, Springer-Verlag Berlin, Heidelberg
2. Hakan Ceylan and Yookyung Kim, 2009, Language Identification of Search Engine Queries, Proceedings of the 47thAnnual Meeting of the ACL and the 4th IJCNLP of the AFNLP, pages 1066–1074, Suntec, Singapore, 2-7 August 2009
3. Abdelmalek Amine, 2010, Automatic Language Identification: An AlterNaive Unsupervised Approach Using A New Hybrid Algorithm, International Journal of Computer Science and Applications, Technomathematics Research Foundation, Vol. 7, No. 1, pp. 94 – 107, 2010
4. Radim Rehurek and Milan Kolkus, 2009, Language Identification on the Web: Extending the Dictionary Method, A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 357–368, 2009.
5. Bethlehem Mengistu, 2002, N-gram-Based Automatic Indexing for Amharic Text, Addis Ababa University, pp. 357–368, 2002.
6. Kidst Ergetie Andargie, 2017, General Purpose Language Identification for Ethio-Semitic Languages Using Hybrid Approach, Jimma Institute Of Technology, Department Of Information Technology, 2017
7. Bashir Ahmed, Sung-Hyuk Cha, and Charles Tappert, Language Identification from Text Using N-gram Based Cumulative Frequency Addition, Faculty Research Day, CSIS, Pace University, May 7th, 2004
8. Leonid Panich, Comparison of Language Identification Techniques, Institut für Informatik Datenbanken und Informations systeme, June, 2015, Page 3
9. Lena Grothe, Ernesto William De Luca and Andreas Nurnberger, A Comparative Study on Language Identification Methods, University of Magdeburg, Faculty of Computer Science 39106, Magdeburg, Germany, December, 2008
10. Thomas Gottron1 and Nedim Lipka, A Comparison of Language Identification Approaches on Short, Query-Style Texts, 1 Institut für Informatik, Johannes Gutenberg-Universität Mainz, 55099 Mainz, Germany, December 2009

11. Archana Garg, A Survey of Language Identification Techniques and Applications, Department of Computer Science and Applications, Panjab University, Chandigarh, India, December 2014, page 1.
12. Ted Dunning, Statistical Identification of Language, Computing Research Laboratory, New Mexico State University, March 10, 1994
13. Bizuneh Mamuye Birhan, The Application of Websom for Amharic Text Retrieval, Addis Ababa University, School Of Graduate Studies, Faculty Of Informatics, Department Of Information Science, July 2003
14. N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers," Machine learning, vol. 29, no. 2-3, pp. 131–163, 1997
15. Daisuke Okanohara, Jun'ichi Tsujii, "Text Categorization with All Substring Features, ASCII Media Works, 2008
16. Paul McNamee, "Language Identification: A Solved Problem Suitable For Undergraduate Instruction", Journal of Computing Sciences in Colleges archive, Volume 20 Issue 3, February 2005**,** Pages 94-101
17. Prager, John M.,"Linguini: Language Identification for Multilingual Documents". In: Journal of Management Information Systems, 1999, S. 1–11
18. Poutsma, Arjen," Applying Monte Carlo Techniques to Language Identification". Computational Linguistics in the Netherlands 2001, Language and Computers, Volume: 45, 2016
19. Cavnar, William B. and Trenkle, John M., "N-Gram-Based Text Categorization. , In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval, 1994, S. 161–175
20. P. Sibun and J.C. Reynar. 1996. Language identification: Examining the issues. In 5th Symposium on Document Analysis and Information Retrieval, pages 125–135, Las Vegas, Nevada, U.S.A.

APPENDICES

Appendix 1: The Standard Unicode for Ethiopic

The Unicode Standard, Version 11.0 taken from https://unicode.org/charts/PDF/U1200.pdf

Ethiopic Syllable

UTF	Glyph	Ethiopic Syllable
1200	ሀ	HA
1201	ሁ	HU
1202	ሂ	HI
1203	ሃ	HAA
1204	ሄ	HEE
1205	ህ	HE
1206	ሆ	HO
1207	ሇ	HOA
1208	ለ	LA
1209	ሉ	LU
120A	ሊ	LI
120B	ላ	LAA
120C	ሌ	LEE
120D	ል	LE
120E	ሎ	LO
120F	ሏ	LWA
1210	ሐ	HHA
1211	ሑ	HHU
1212	ሒ	HHI
1213	ሓ	HHAA
1214	ሔ	HHEE
1215	ሕ	HHE
1216	ሖ	HHO
1217	ሗ	HHWA
1218	መ	MA
1219	ሙ	MU
121A	ሚ	MI
121B	ማ	MAA
121C	ሜ	MEE
121D	ም	ME
121E	ሞ	MO
121F	ሟ	MWA
1220	ሠ	SZA

UTF	Glyph	Ethiopic Syllable
1221	ሡ	SZU
1222	ሢ	SZI
1223	ሣ	SZAA
1224	ሤ	SZEE
1225	ሥ	SZE
1226	ሦ	SZO
1227	ሧ	SZWA
1228	ረ	RA
1229	ሩ	RU
122A	ሪ	RI
122B	ራ	RAA
122C	ሬ	REE
122D	ር	RE
122E	ሮ	RO
122F	ሯ	RWA
1230	ሰ	SA
1231	ሱ	SU
1232	ሲ	SI
1233	ሳ	SAA
1234	ሴ	SEE
1235	ስ	SE
1236	ሶ	SO
1237	ሷ	SWA
1238	ሸ	SHA
1239	ሹ	SHU
123A	ሺ	SHI
123B	ሻ	SHAA
123C	ሼ	SHEE
123D	ሽ	SHE
123E	ሾ	SHO
123F	ሿ	SHWA
1240	ቀ	QA
1241	ቁ	QU

UTF	Glyph	Ethiopic Syllable
1242	ቂ	QI
1243	ቃ	QAA
1244	ቄ	QEE
1245	ቅ	QE
1246	ቆ	QO
1247	ቇ	QOA
1248	ቈ	QWA
1249	<reserved>	<reserved>
124A	ቊ	QWI
124B	ቋ	QWAA
124C	ቌ	QWEE
124D	ቍ	QWE
124E	<reserved>	<reserved>
124F	<reserved>	<reserved>
1250	ቐ	QHA
1251	ቑ	QHU
1252	ቒ	QHI
1253	ቓ	QHAA
1254	ቔ	QHEE
1255	ቕ	QHE
1256	ቖ	QHO
1257	<reserved>	<reserved>
1258	ቘ	QHWA
1259	<reserved>	<reserved>
125A	ቚ	QHWI
125B	ቛ	QHWAA
125C	ቜ	QHWEE
125D	ቝ	QHWE
125E	<reserved>	<reserved>
125F	<reserved>	<reserved>
1260	በ	BA
1261	ቡ	BU
1262	ቢ	BI
1263	ባ	BAA
1264	ቤ	BEE
1265	ብ	BE
1266	ቦ	BO
1267	ቧ	BWA
1268	ቨ	VA
1269	ቩ	VU
126A	ቪ	VI
126B	ቫ	VAA

UTF	Glyph	Ethiopic Syllable
126C	ቬ	VEE
126D	ቭ	VE
126E	ቮ	VO
126F	ቯ	VWA
1270	ተ	TA
1271	ቱ	TU
1272	ቲ	TI
1273	ታ	TAA
1274	ቴ	TEE
1275	ት	TE
1276	ቶ	TO
1277	ቷ	TWA
1278	ቸ	CA
1279	ቹ	CU
127A	ቺ	CI
127B	ቻ	CAA
127C	ቼ	CEE
127D	ች	CE
127E	ቾ	CO
127F	ቿ	CWA
1280	ኀ	XA
1281	ኁ	XU
1282	ኂ	XI
1283	ኃ	XAA
1284	ኄ	XEE
1285	ኅ	XE
1286	ኆ	XO
1287	ኇ	XOA
1288	ኈ	XWA
1289	<reserved>	<reserved>
128A	ኊ	XWI
128B	ኋ	XWAA
128C	ኌ	XWEE
128D	ኍ	XWE
128E	<reserved>	<reserved>
128F	<reserved>	<reserved>
1290	ነ	NA
1291	ኑ	NU
1292	ኒ	NI
1293	ና	NAA
1294	ኔ	NEE
1295	ን	NE

UTF	Glyph	Ethiopic Syllable
1296	ኖ	NO
1297	ኗ	NWA
1298	ኘ	NYA
1299	ኙ	NYU
129A	ኚ	NYI
129B	ኛ	NYAA
129C	ኜ	NYEE
129D	ኝ	NYE
129E	ኞ	NYO
129F	ኟ	NYWA
12A0	አ	GLOTTAL
12A1	ኡ	GLOTTAL
12A2	ኢ	GLOTTAL
12A3	ኣ	GLOTTAL
12A4	ኤ	GLOTTAL
12A5	እ	GLOTTAL
12A6	ኦ	GLOTTAL
12A7	ኧ	GLOTTAL
12A8	ከ	KA
12A9	ኩ	KU
12AA	ኪ	KI
12AB	ካ	KAA
12AC	ኬ	KEE
12AD	ክ	KE
12AE	ኮ	KO
12AF	ኰ	KOA
12B0	ኲ	KWA
12B1	<reserved>	<reserved>
12B2	ኳ	KWI
12B3	ኴ	KWAA
12B4	ኵ	KWEE
12B5	ኸ	KWE
12B6	<reserved>	<reserved>
12B7	<reserved>	<reserved>
12B8	ኹ	KXA
12B9	ኺ	KXU
12BA	ኻ	KXI
12BB	ኼ	KXAA
12BC	ኽ	KXEE
12BD	ኾ	KXE
12BE	ዀ	KXO
12BF	<reserved>	<reserved>

UTF	Glyph	Ethiopic Syllable
12C0	ዂ	KXWA
12C1	<reserved>	<reserved>
12C2	ዃ	KXWI
12C3	ዄ	KXWAA
12C4	ዅ	KXWEE
12C5	ወ	KXWE
12C6	<reserved>	<reserved>
12C7	<reserved>	<reserved>
12C8	ዐ	WA
12C9	ዑ	WU
12CA	ዒ	WI
12CB	ዓ	WAA
12CC	ዔ	WEE
12CD	ዕ	WE
12CE	ዖ	WO
12CF	ዏ	WOA
12D0	ዐ	PHARYNGEAL
12D1	ዑ	PHARYNGEAL
12D2	ዒ	PHARYNGEAL
12D3	ዓ	PHARYNGEAL
12D4	ዔ	PHARYNGEAL
12D5	ዕ	PHARYNGEAL
12D6	ዖ	PHARYNGEAL
12D7	<reserved>	<reserved>
12D8	ዘ	ZA
12D9	ዙ	ZU
12DA	ዚ	ZI
12DB	ዛ	ZAA
12DC	ዜ	ZEE
12DD	ዝ	ZE
12DE	ዞ	ZO
12DF	ዟ	ZWA
12	ዠ	ZHA
120	ዡ	ZHU
1200	ዢ	ZHI
12000	ዣ	ZHAA
120000	ዤ	ZHEE
1200000	ዥ	ZHE
12000000	ዦ	ZHO
1.2E+08	ዧ	ZHWA
1.2E+09	የ	YA
1.2E+10	ዩ	YU

UTF	Glyph	Ethiopic Syllable
12EA	ዪ	YI
12EB	ያ	YAA
12EC	ዬ	YEE
12ED	ይ	YE
12EE	ዮ	YO
12EF	ዯ	YOA
12F0	ደ	DA
12F1	ዱ	DU
12F2	ዲ	DI
12F3	ዳ	DAA
12F4	ዴ	DEE
12F5	ድ	DE
12F6	ዶ	DO
12F7	ዷ	DWA
12F8	ዸ	DDA
12F9	ዹ	DDU
12FA	ዺ	DDI
12FB	ዻ	DDAA
12FC	ዼ	DDEE
12FD	ዽ	DDE
12FE	ዾ	DDO
12FF	ዿ	DDWA
1300	ጀ	JA
1301	ጁ	JU
1302	ጂ	JI
1303	ጃ	JAA
1304	ጄ	JEE
1305	ጅ	JE
1306	ጆ	JO
1307	ጇ	JWA
1308	ገ	GA
1309	ጉ	GU
130A	ጊ	GI
130B	ጋ	GAA
130C	ጌ	GEE
130D	ግ	GE
130E	ጎ	GO
130F	ጏ	GOA
1310	ጐ	GWA
1311	<reserved>	<reserved>
1312	ጒ	GWI
1313	ጓ	GWAA

UTF	Glyph	Ethiopic Syllable
1314	ጔ	GWEE
1315	ጕ	GWE
1316	<reserved>	<reserved>
1317	<reserved>	<reserved>
1318	ጘ	GGA
1319	ጙ	GGU
131A	ጚ	GGI
131B	ጛ	GGAA
131C	ጜ	GGEE
131D	ጝ	GGE
131E	ጞ	GGO
131F	ጟ	GGWAA
1320	ጠ	THA
1321	ጡ	THU
1322	ጢ	THI
1323	ጣ	THAA
1324	ጤ	THEE
1325	ጥ	THE
1326	ጦ	THO
1327	ጧ	THWA
1328	ጨ	CHA
1329	ጩ	CHU
132A	ጪ	CHI
132B	ጫ	CHAA
132C	ጬ	CHEE
132D	ጭ	CHE
132E	ጮ	CHO
132F	ጯ	CHWA
1330	ጰ	PHA
1331	ጱ	PHU
1332	ጲ	PHI
1333	ጳ	PHAA
1334	ጴ	PHEE
1335	ጵ	PHE
1336	ጶ	PHO
1337	ጷ	PHWA
1338	ጸ	TSA
1339	ጹ	TSU
133A	ጺ	TSI
133B	ጻ	TSAA
133C	ጼ	TSEE
133D	ጽ	TSE

UTF	Glyph	Ethiopic Syllable
133E	ጸ	TSO
133F	ጹ	TSWA
1340	ፀ	TZA
1341	ፁ	TZU
1342	ፂ	TZI
1343	ፃ	TZAA
1344	ፄ	TZEE
1345	ፅ	TZE
1346	ፆ	TZO
1347	ፇ	TZOA
1348	ፈ	FA
1349	ፉ	FU
134A	ፊ	FI
134B	ፋ	FAA
134C	ፌ	FEE
134D	ፍ	FE
134E	ፎ	FO
134F	ፏ	FWA
1350	ፐ	PA
1351	ፑ	PU
1352	ፒ	PI
1353	ፓ	PAA
1354	ፔ	PEE
1355	ፕ	PE
1356	ፖ	PO
1357	ፗ	PWA
1358	ፘ	RYA
1359	ፙ	MYA
135A	ፚ	FYA

Punctuation

UTF	Glyph	Ethiopic Punctuation
1360	፠	SECTION MARK
1361	፡	WORDSPACE
1362	።	FULL STOP
1363	፣	COMMA
1364	፤	SEMICOLON
1365	፥	COLON
1366	፦	PREFACE COLON
1367	፧	QUESTION MARK
1368	፨	PARAGRAPH SEPARATOR

Digits

UTF	Glyph	Ethiopic Digit
1369	፩	ONE
136A	፪	TWO
136B	፫	THREE
136C	፬	FOUR
136D	፭	FIVE
136E	፮	SIX
136F	፯	SEVEN
1370	፰	EIGHT
1371	፱	NINE

Numbers

UTF	Glyph	Ethiopic Number
1372	፲	TEN
1373	፳	TWENTY
1374	፴	THIRTY
1375	፵	FORTY
1376	፶	FIFTY
1377	፷	SIXTY
1378	፸	SEVENTY
1379	፹	EIGHTY
137A	፺	NINETY
137B	፻	HUNDRED THOUSAND
137C	፼	TEN

Appendix 2: Sample Data

Below is the sample text used for training and the full corpus and project code is available on https://github.com/Rediat/cfa.git in corpus\rawSource folder.

Amharic

1የዳዊት ልጅ፥ የአብርሃም ልጅ የጌታችን የኢየሱስ ክርስቶስ የትውልድ መጽሐፍ፤ 2አብርሃም ይስሐቅን ወለደ፤ ይስሐቅም ያዕቆብን ወለደ፤ ያዕቆብም ይሁዳንና ወንድሞቹን ወለደ። 3ይሁዳም ከትእማር ፋሬስንና ዛራን ወለደ፤ ፋሬስም ኤስሮምን ወለደ፤ ኤስሮምም አራምን ወለደ። 4አራምም አሚናዳብን ወለደ፤ አሚናዳብም ነአሶንን ወለደ፤ ነአሶንም ሰልሞንን ወለደ። 5ሰልሞንም ከራኬብ ቦኤዝን ወለደ፤ ቦኤዝም ከሩት ኢዮቤድን ወለደ፤ ኢዮቤድም እሴይን ወለደ። 6እሴይም ንጉሥ ዳዊትን ወለደ። 7ንጉሥ ዳዊትም ከኦርዮ ሚስት ሰሎሞንን ወለደ፤ ሰሎሞንም ሮብዓምን ወለደ፤ ሮብዓምም አብያን ወለደ፤ አብያም አሳፍን ወለደ። 8አሳፍም ኢዮሳፍጥን ወለደ፤ ኢዮሳፍጥም ኢዮራምን ወለደ፤ ኢዮራምም ዖዝያንን ወለደ። 9ዖዝያንም ኢዮአታምን ወለደ፤ ኢዮአታምም አካዝን ወለደ፤ አካዝም ሕዝቅያስን ወለደ። 10ሕዝቅያስም ምናሴን ወለደ፤ ምናሴም አሞጽን ወለደ፤ አሞጽም ኢዮስያስን ወለደ። 11ኢዮስያስም በባቢሎን ምርኮ ጊዜ ኢኮንያንንና ወንድሞቹን ወለደ። 12ከባቢሎን ምርኮም በኋላ ኢኮንያን ሰላትያልን ወለደ፤ ሰላትያልም ዘሩባቤልን ወለደ።

Geez

በቀዳሚ ገብረ እግዚአብሔር ሰማየ ወምድረ ወምድርሰ ኢታስተርኢ ወኢኮነት ድሉተ ወጽልሞት መልዕልተ ቀላይ ወመንፈሰ እግዚአብሔር ይጼልል መልዕልተ ማይ ወይቤ እግዚአብሔር ለይኩን ብርሃን ወኮነ ብርሃን ወርእዮ እግዚአብሔር ለብርሃን ከመ ሠናይ ወፈለጠ እግዚአብሔር ማእከለ ብርሃን ወማእከለ ጽልሞት ወሰመዮ እግዚአብሔር ለብርሃን ዕለተ ወለጽልሞት ሌሊተ ወኮነ ሌሊተ ወጸብሐ ወኮነ መዓልተ ፩ ወይቤ እግዚአብሔር ለይኩን ጠፈር ማእከለ ማይ ከመ ይፍልጥ ማእከለ ማይ ወኮነ ከማሁ ወገብረ እግዚአብሔር ጠፈረ ወፈለጠ እግዚአብሔር ማእብለ ማይ ዘታሕተ ጠፈር ወማእከለ ማይ ዘመልዕልተ ጠፈር ወሰመዮ እግዚአብሔር ለውእቱ ጠፈር ሰማየ ወርእየ እግዚአብሔር ከመ ሠናይ ወኮነ ሌሊተ ወጸብሐ ወኮነ ካልእተ ዕለተ ወይቤ እግዚአብሔር ለይትጋባእ ማይ ዘመትሕተ ሰማይ ውስተ

Guragigna

1የዳዊትም የአብርኻምም ዘር የኸረ ኢየሱስ ክርስቶስ የትጨነወ ዘር ዝኸታው፡- 2(16) ክርስቶስ የዋሪ ኢየሱስ የጨነች፥ ማርያም የፈቸ ዮሴፍ የያእቆብ ኧርቹ፤ 3(15) ያእቆብ የማታን ማታን የአልአዛር አልአዛር የኤሉድ 4(14) ኤልዩድ የአኪም አኪም የሳዶቅ ሳዶቅ የአዞር 5(13) አዞር የኤልያቄም ኤልያቄም የአብዩድ አብዩድ የዘሩባቤል 6(12) ዘሩባቤል የሰላትያል ሰላትያል የእስራኤል ሰብ በባቢሎን ሰብ ባቢሎን ገነ በማነኹዮ አንቅ ተኢኮንያን ተጨነም። 7(11) ኢኮንያን ጐፔመታ የእስራኤል ሰብ ባቢሎን ገነ ማነኹም ቦሰጅዮ ዘበር ተኢዮስያስ ተጨነቦም። 8(10) ኢዮስያስ የአሞን አሞን የምናሴ ምናሴ የኽዝቅያስ 9ኽዝቅያስ የአካዝ አካዝ የኢዮአታም ኢዮአታም የኡዝያ 10(8) ኡዝያ የኢዮራም ኢዮራም የኢየሳፍጥ ኢዮሳፍጥ የአሳፍ 11(7) አሳፍ የአቢያ አቢያ የሮብአም ሮብአም የሰለሞን 12(6) ሰለሞን ተኦርዮ ምሸት

Tigrigna

ወዲ ዳዊት ወዲ ኣብርሃም ናይ ዝኾነ ናይ ኢየሱስ ክርስቶስ ትውልዲ መጽሓፍ፡- 2ኣብርሃም ንይስሓቅ ወለደ፤ ይስሓቅ ንያዕቆብ ወለደ፤ ያዕቆብ ንይሁዳን ነኀዋቱን ወለደ 3ይሁዳ ኽዓ ካብ ትእማር ንፋሬስን ንዛራን ወለደ ፋሬስ ንኤስሮም ወለደ ኤስሮም ንኣራም ወለደ 4ኣራም ንኣሚናዳብ ወለደ ኣሚናዳብ ንነኣሶን ወለደ ነኣሶን ንሰልሞን ወለደ 5ሰልሞን ካብ ረኣብ ንቦኤዝ ወለደ ቦኤዝ ካብ ሩት ንኢዮቤድ ወለደ ኢዮቤድ ንእሴይ ወለደ 6እሴይ ንንጉሥ ዳዊት ወለደ። ንጉሥ ዳዊት ካብ ሰበይቲ ኦርዮ ንሰሎሞን ወለደ 7ሰሎሞን ንሮብኣም ወለደ ሮብኣም ንኣብያ ወለደ ኣብያ ንኣሳፍ ወለደ 8ኣሳፍ ንኢዮሳፍጥ ወለደ ኢዮሳፍጥ ንኢዮራም ወለደ ኢዮራም ንዖዝያ ወለደ 9ዖዝያ ንኢየኣታም ወለደ ኢዮኣታም ንኣካዝ ወለደ ኣካዝ ንሕዝቅያስ ወለደ 10ሕዝቅያስ ንምናሴ ወለደ ምናሴ ንኣሞጽ ወለደ ኣሞጽ ንኢዮስያስ ወለደ 11ኢዮስያስ ከኣ ብጊዜ ምርኮ ባቢሎን ንኢኮንያንን ነኀዋቱን ወለደ። 12ድኅሪ ምርኮ ባቢሎን ድማ ኢኮንያን ንሰላትያል ወለደ ሰላትያል ንዘሩባቤል ወለደ

Appendix 3: Code Samples

The sample code of the Modelling Function that generates five different character based n-gram features is taken from modellerAll.py file. The complete project code is available on https://github.com/Rediat/cfa.git.

```
import os ; import sys ; import re ; import glob
import string ; import datetime ; import langid as l

def modeler(started, mod,wordbased=0,location=0,infinity=0):

	print ('\n{}'.format('='*110))
	print ('\n{} - Loading corpus files to memory and generating models
...'.format(datetime.datetime.now()))

	files = '*.txt'
	mostfrequent = {} #the most frequent ngram in the corpus
	wordcount = {} ; vocabulary = {}
	language = dict(am='Amharic',ge='Geez',gu='Guragigna',ti='Tigrigna')
	maxg = 5

	started = datetime.datetime.now()
	myfile=mod+'.txt'
	for ct in range(1,11):
		path = 'corpus/training/'+str(ct)+'/' ; model =[]
		print ('\n{}\nProcessing corpus {}\n{}'.format('='*110,ct,'-'*110))
		for infile in glob.glob(os.path.join(path, files)):
			try:

				#Extract the file name
				filename = infile.split('/')[-1]
				lang = filename[:2]

				#Open and read file from corpus
				f=open(infile,'r')#, encoding = 'utf8' )
				raw = f.read()
				rawtext = [lang,raw]
				f.close()

				#Word/Ngram level parsing
				ngrams=[]

				'''\nSelect Model type number below: \n\n
				1. The Model is based on Fixed Length N-grams without
location features - Baseline [bl]. \n
				2. The Model is based on source text - Byte order N-grams
[by]. \n
				3. The Model is based on Fixed Length N-grams with
location features [fl]. \n
				4. The Model is based on Infiniti-grams without location
features [in]. \n
				5. The Model is based on Infiniti-grams with location
features [il]. \n
				6. The Model is based on Word Frequency without location
features [wr]. \n
```

```
                        7. Exit.:   '))
                        '''
                        if mod=='bl': #to generate ngrams from source text
wordbased baseline
                              temp = l.regex(rawtext)[1].split()
                              wordlist = []
                              for i in temp:
                                    wordlist.extend(l.ngram([lang,i])[1])
                              ngrams=[lang,wordlist,mod]

                        elif mod=='by': #to generate ngrams from source text fixed
byteorder
                              ngrams = l.ngram(l.regex(rawtext))
                              ngrams.append(mod)

                        elif mod=='fl': #to generate ngrams model (inifinigram)
from words taken from source files
                              temp = l.regex(rawtext)[1].split()
                              wordlist = []
                              for i in temp:

      wordlist.extend(l.ngram([lang,i],0,location,infinity)[1])
                              ngrams=[lang,wordlist,mod]

                        elif mod=='il' or mod=='in': #to generate ngrams model
(inifinigram) from words taken from source files
                              temp = l.regex(rawtext)[1].split()
                              wordlist = []
                              for i in temp:
                                    maxg = len(i) if maxg <   len(i) else maxg
                                    if location==0:

      wordlist.extend(l.ngram([lang,i],0,location,infinity)[1])
                                    else:

      wordlist.extend(l.ngram([lang,i],0,location,infinity)[1])
                              ngrams=[lang,wordlist,mod]

                        elif mod=='wr':
                              temp = l.regex(rawtext)[1].split()
                              ngrams = [lang,temp,mod]

                        print('\t{} - Completed removing punctuation marks and
numbers for {} language'.format(datetime.datetime.now(),language[lang]))
                        summary =
l.summerize(model,l.wordgrams(ngrams),mostfrequent,wordcount,vocabulary)

                        print('\t{} - Completed building sorted frequency
distribution for {} language'.format(datetime.datetime.now(),language[lang]))
                        print('{}{}'.format('\t','-'*100))

                  except IOError:
                        print ('Error: Can not open the file: ',lang)
                        return
            print('{} - Saving the model for all languages to
models/{}/{}'.format(datetime.datetime.now(),ct,myfile))
            l.savetofile(summary,started,mod,maxg,ct)
```

Appendix 3: Model Sample

The language model has the columns Language, N-gram, number of characters, frequency, bytes, Overall frequency weight for CFA, Cumulative of Relative Frequency, and Percentage considered.

Below is the sample taken from Fixed Length Character n-gram with location features generated from a Bag of words from model 3.

The complete files are available on https://github.com/Rediat/cfa.git inside models folder.

```
ti,___□,4,1122,82,1.000003830618333,0.00538822083061201,1
ti,__□,3,1122,80,1.000003830618333,0.01077644166122402,1
ti,_□,2,1122,78,1.000003830618333,0.01616466249183603,1
ti,____□,5,1122,84,1.000003830618333,0.02155288332244804,1
ti,□___,4,951,82,1.0000032468075175,0.026119904721656612,1
ti,□_,2,951,78,1.0000032468075175,0.03068692612086519,1
ti,□____,5,951,84,1.0000032468075175,0.03525394752007376,1
ti,□__,3,951,80,1.0000032468075175,0.039820968919282336,1
ti,___□,4,823,82,1.0000028098029305,0.043773291328902376,1
ti,__□,3,823,80,1.0000028098029305,0.04772561373852242,1
ti,____□,5,823,84,1.0000028098029305,0.05167793614814246,1
ti,_□,2,823,78,1.0000028098029305,0.0556302585577625,1
ti,□_,2,719,78,1.0000024547367037,0.05908313803834185,1
ti,□___,4,719,82,1.0000024547367037,0.0625360175189212,1
ti,□____,5,719,84,1.0000024547367037,0.06598889699950056,1
ti,□__,3,719,80,1.0000024547367037,0.06944177648007992,1
ti,□___,4,514,82,1.0000017548465447,0.07191017711014637,1
ti,□__,3,514,80,1.0000017548465447,0.07437857774021284,1
ti,□_,2,514,78,1.0000017548465447,0.0768469783702793,1
ti,□____,5,514,84,1.0000017548465447,0.07931537900034577,1
ti,□_,2,435,78,1.000001485132776,0.08140439509777556,1
ti,□____,5,435,84,1.000001485132776,0.08349341119520534,1
ti,□___,4,435,82,1.000001485132776,0.08558242729263514,1
ti,□__,3,435,80,1.000001485132776,0.08767144339006493,1
ti,_□,2,434,78,1.0000014817186778,0.08975565715163855,1
ti,____□,5,434,84,1.0000014817186778,0.09183987091321219,1
ti,___□,4,434,82,1.0000014817186778,0.09392408467478582,1
ti,__□,3,434,80,1.0000014817186778,0.09600829843635944,1
ti,□____,5,418,84,1.0000014270931046,0.09801567482423451,1
ti,□__,3,418,80,1.0000014270931046,0.10002305121210957,1
ti,□_,2,418,78,1.0000014270931046,0.10203042759998464,1
ti,□___,4,418,82,1.0000014270931046,0.1040378039878597,1
ti,__□,3,368,80,1.0000012563881877,0.10580506358292674,1
ti,____□,5,368,84,1.0000012563881877,0.10757232317799377,1
ti,_□,2,368,78,1.0000012563881877,0.10933958277306081,1
ti,___□,4,368,82,1.0000012563881877,0.11110684236812786,1
ti,□___,4,356,82,1.0000012154190077,0.11281647393292098,1
ti,□_,2,356,78,1.0000012154190077,0.11452610549771408,1
ti,□__,3,356,80,1.0000012154190077,0.1162357370625072,1
ti,□____,5,356,84,1.0000012154190077,0.11794536862730032,1
ti,□_,2,345,78,1.000001177863926,0.11960217449767567,1
ti,□__,3,345,80,1.000001177863926,0.12125898036805102,1
ti,□____,5,345,84,1.000001177863926,0.12291578623842637,1
ti,□___,4,345,82,1.000001177863926,0.12457259210880173,1
ti,___□,4,341,82,1.0000011642075326,0.12621018863575242,1
```

Appendix 4: Test strings sample

The columns are Language, Test phrase, words, characters, and bytes. Below is a sample taken from the test phrases for the test number 5.

The complete files are available on https://github.com/Rediat/cfa.git in the samples folder.

```
am,ለመከሩ,1,4,82
am,ለመከሩ ሠራተኛን,2,10,94
am,ሠራተኛን,1,5,84
am,ለመከሩ ሠራተኛን ይጨምር,3,15,104
am,ሠራተኛን ይጨምር,2,10,94
am,ይጨምር,1,4,82
am,ለመከሩ ሠራተኛን ይጨምር ዘንድ,4,19,112
am,ሠራተኛን ይጨምር ዘንድ,3,14,102
am,ይጨምር ዘንድ,2,8,90
am,ዘንድ,1,3,80
am,ለመከሩ ሠራተኛን ይጨምር ዘንድ የመከሩን,5,25,124
am,ሠራተኛን ይጨምር ዘንድ የመከሩን,4,20,114
am,ይጨምር ዘንድ የመከሩን,3,14,102
am,ዘንድ የመከሩን,2,9,92
am,የመከሩን,1,5,84
am,ለመከሩ ሠራተኛን ይጨምር ዘንድ የመከሩን ባለቤት,6,30,134
am,ሠራተኛን ይጨምር ዘንድ የመከሩን ባለቤት,5,25,124
am,ይጨምር ዘንድ የመከሩን ባለቤት,4,19,112
am,ዘንድ የመከሩን ባለቤት,3,14,102
am,የመከሩን ባለቤት,2,10,94
am,ባለቤት,1,4,82
am,ለመከሩ ሠራተኛን ይጨምር ዘንድ የመከሩን ባለቤት ለምኑት,7,35,144
am,ሠራተኛን ይጨምር ዘንድ የመከሩን ባለቤት ለምኑት,6,30,134
am,ይጨምር ዘንድ የመከሩን ባለቤት ለምኑት,5,24,122
am,ዘንድ የመከሩን ባለቤት ለምኑት,4,19,112
am,የመከሩን ባለቤት ለምኑት,3,15,104
am,ባለቤት ለምኑት,2,9,92
am,ለምኑት,1,4,82
am,ለመከሩ ሠራተኛን ይጨምር ዘንድ የመከሩን ባለቤት ለምኑት ዐሥራ,8,39,152
am,ሠራተኛን ይጨምር ዘንድ የመከሩን ባለቤት ለምኑት ዐሥራ,7,34,142
am,ይጨምር ዘንድ የመከሩን ባለቤት ለምኑት ዐሥራ,6,28,130
am,ዘንድ የመከሩን ባለቤት ለምኑት ዐሥራ,5,23,120
am,የመከሩን ባለቤት ለምኑት ዐሥራ,4,19,112
am,ባለቤት ለምኑት ዐሥራ,3,13,100
am,ለምኑት ዐሥራ,2,8,90
am,ዐሥራ,1,3,80
am,ለመከሩ ሠራተኛን ይጨምር ዘንድ የመከሩን ባለቤት ለምኑት ዐሥራ ሁለቱን,9,44,162
am,ሠራተኛን ይጨምር ዘንድ የመከሩን ባለቤት ለምኑት ዐሥራ ሁለቱን,8,39,152
am,ይጨምር ዘንድ የመከሩን ባለቤት ለምኑት ዐሥራ ሁለቱን,7,33,140
am,ዘንድ የመከሩን ባለቤት ለምኑት ዐሥራ ሁለቱን,6,28,130
am,የመከሩን ባለቤት ለምኑት ዐሥራ ሁለቱን,5,24,122
am,ባለቤት ለምኑት ዐሥራ ሁለቱን,4,18,110
am,ለምኑት ዐሥራ ሁለቱን,3,13,100
am,ዐሥራ ሁለቱን,2,8,90
am,ሁለቱን,1,4,82
am,ለመከሩ ሠራተኛን ይጨምር ዘንድ የመከሩን ባለቤት ለምኑት ዐሥራ ሁለቱን ደቀ,10,47,168
am,ሠራተኛን ይጨምር ዘንድ የመከሩን ባለቤት ለምኑት ዐሥራ ሁለቱን ደቀ,9,42,158
am,ይጨምር ዘንድ የመከሩን ባለቤት ለምኑት ዐሥራ ሁለቱን ደቀ,8,36,146
am,ዘንድ የመከሩን ባለቤት ለምኑት ዐሥራ ሁለቱን ደቀ,7,31,136
```

Appendix 5: Classifier Screen Output

Classifier screen output based on Context I

```
* The Model is based on Fixed Length N-grams without location features - Baseline [bl] on test 5. **
Press 1 to classify 2 to change model [0 to exit] :   1
Insert between 1 and 25 to set the test phrase length from testing files   :   1
-------------------------------------------------------------------------------------------------
Average length of test strings: 1 word(s) / 4 character(s) / 82 bytes   Model: 53,807 lines.
=================================================================================================
N-grams          |Observations   |Accuracy       |Precision      |Recall         |F-score
-------------------------------------------------------------------------------------------------
CFA (2,3,4,5)   |2,234          |    0.8532     |    0.8545     |    0.8532     |    0.8539
NBC (2,3,4,5)   |2,234          |    0.8371     |    0.8393     |    0.8378     |    0.8386
-------------------------------------------------------------------------------------------------

*************** The Model is based on source text - Byteorder Ngrams [by] on test 5. ***************
Press 1 to classify 2 to change model [0 to exit] :   1
Insert between 1 and 25 to set the test phrase length from testing files   :   1
-------------------------------------------------------------------------------------------------
Average length of test strings: 1 word(s) / 4 character(s) / 82 bytes   Model: 220,648 lines.
=================================================================================================
N-grams          |Observations   |Accuracy       |Precision      |Recall         |F-score
-------------------------------------------------------------------------------------------------
CFA (2,3,4,5)   |2,234          |    0.8532     |    0.8545     |    0.8532     |    0.8539
NBC (2,3,4,5)   |2,234          |    0.8393     |    0.8395     |    0.8398     |    0.8397
-------------------------------------------------------------------------------------------------

******** The Model is based on Fixed Length N-grams with location features [fl] on test 5. *********
Press 1 to classify 2 to change model [0 to exit] :   1
Insert between 1 and 25 to set the test phrase length from testing files   :   1
-------------------------------------------------------------------------------------------------
Average length of test strings: 1 word(s) / 4 character(s) / 82 bytes   Model: 148,724 lines.
=================================================================================================
N-grams          |Observations   |Accuracy       |Precision      |Recall         |F-score
-------------------------------------------------------------------------------------------------
CFA (2,3,4,5)   |2,234          |    0.8765     |    0.8765     |    0.8766     |    0.8766
NBC (2,3,4,5)   |2,234          |    0.8657     |    0.8661     |    0.8656     |    0.8658
-------------------------------------------------------------------------------------------------
```

Classifier screen output based on Context II

```
********** The Model is based on Infiniti-grams without location features [in] on test 5. **********

Press 1 to classify 2 to change model [0 to exit] :   1
Insert between 1 and 25 to set the test phrase length from testing files   :   5

Average length of test strings: 5 word(s) / 23 character(s) / 120 bytes
=============================================================================================
N-grams          |Observations   |Accuracy       |Precision      |Recall         |F-score
---------------------------------------------------------------------------------------------
CFA 2            |3,101          |     94.36%    |     94.60%    |     94.31%    |     94.46%
CFA 3            |3,101          |     92.21%    |     92.73%    |     91.91%    |     92.32%
CFA 4            |3,101          |     90.27%    |     89.95%    |     88.00%    |     88.97%
CFA 5            |3,101          |     83.36%    |     82.86%    |     78.25%    |     80.49%
---------------------------------------------------------------------------------------------
CFA (2,3,4,5)    |12,404         |     90.05%    |     90.04%    |     88.12%    |     89.06%
---------------------------------------------------------------------------------------------

Average length of test strings: 5 word(s) / 23 character(s) / 120 bytes
=============================================================================================
N-grams          |Observations   |Accuracy       |Precision      |Recall         |F-score
---------------------------------------------------------------------------------------------
NBC 2            |3,101          |     97.29%    |     97.28%    |     97.25%    |     97.27%
NBC 3            |3,101          |     91.20%    |     91.68%    |     90.71%    |     91.20%
NBC 4            |3,101          |     86.06%    |     84.79%    |     84.20%    |     84.49%
NBC 5            |3,101          |     80.06%    |     80.16%    |     75.64%    |     77.83%
---------------------------------------------------------------------------------------------
NBC (2,3,4,5)    |12,404         |     88.65%    |     88.48%    |     86.95%    |     87.70%
---------------------------------------------------------------------------------------------
```

ASSURANCE SHEET

The undersigned agree to accept responsibility for the scientific, ethical and technical conduct of the research and declare that this thesis is my original work and has not been presented for a degree in any other university, and that all source of materials used for the thesis have been duly acknowledged.

Candidate:

_Rediat Bekele Asfaw (Mr.) _____ Signature _____ Date _____

(Name and Signature of candidate)

Advisor:

_Solomon Teferra Abate (PhD)_____ Signature _____ Date _____

(Name and Signature of Advisor)

Examiner:

_Dereje Teferi (PhD)_____ Signature _____ Date _____

(Name and Signature of Examiner)

Examiner:

_Martha Yifiru Tachbelie (PhD)_____ Signature _____ Date _____

(Name and Signature of Examiner)

Chairman:

_____ Signature _____ Date _____

(Name and Signature of Chairman)