Addis Ababa University
College of Natural sciences

Development of Tigrigna grammar checker using hybrid approach

ABRHA GEBREKIROS GEBREEGZAABHER

A THESIS SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE IN PARTIAL FULFILLMENT FOR THE DEGREE OF MASTERS OF SCIENCE IN COMPUTER SCIENCE

ADDIS ABABA, ETHIOPIA

JUNE 2018

ADDIS ABABA UNIVERSITY

COLLEGE OF NATURAL SCIENCES

ABRHA GEBREKIROS GEBREEGZAABHER

ADVISOR: YAREGAL ASSABIE (PhD)

This is to certify that the thesis prepared by **ABRHA GEBREKIROS GEBREEGZAABHER**, titled: ***DEVELOPMENT OF TIGRIGNA GRAMMAR CHECKER USING HYBRID APPROACH*** and submitted in partial fulfillment of the requirements for the Degree of Master of Science in Computer Science complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the Examining committee:

	Name	**Signature**	**Date**
Advisor:			
Examiner:			
Examiner:			

Abstract

Grammar checker is one of the Natural language processing studies that is used to determine whether users' input text is grammatically correct or not. These days, human beings produce lots of texts to communicate through mobile, internet applications, etc. To meet the objective of the communication, we require grammatically correct texts. It can be hard to manually check grammatical error in the text without delay. It also leads to unwanted expenses like time and cost. Thus, to solve the problem several grammar checker systems have been developed for several languages such as Amharic, Afaan-Oromo, English, Swedish, and others. However, when we come to Tigrigna language there is no study attempted to develop grammar checker. Thus, the main objective of this thesis is to present the development of Tigrigna grammar checker.

The proposed Tigrigna grammar checker system is developed using a hybrid of statistical and rule based approach to identify grammar errors in Tigrigna sentences by categorizing grammar errors within a sentence into five grammar errors such as Subject-Verb Agreement, Object-Verb Agreement, Adverb-Verb Agreement, Noun-Modifier Agreement, and Word Sequence Agreement. The system has five main modules namely Preprocessing, Tag sequence gathering, Rule based grammar checker, Statistical grammar checker, and Grammar error filtering. The preprocessing module is used to tokenize and tag tokens within the input text sentence. In the tag sequence gathering module we perform tag splitting, tag sequence extraction, and tag sequence probability calculation to fill the language model with statistical data of the Tigrigna corpus. In the rule based grammar checker module, Subject-Verb Agreement, Object-Verb Agreement, Adverb-Verb Agreement and Noun-Modifier Agreement, grammatical error of the input text sentence is identified by matching patterns against the manually hand written 114 agreement grammatical rules. In the statistical grammar checker module, the WSA grammar error of the input text sentence is determined according to unique tag sequence and probability language model.

We implement the system using C# programming language, SharpNLP as tools and 4645 part of speech annotated sentences as corpus. The evaluation of the system is conducted in four experiments using manually prepared 300 grammatical correct and incorrect sentences. From the

experiments result, the system performs average results of 87.9% precision, 87.5% recall, and 87.6% f-measure.

Keywords: Grammar Checker, Natural Language Processing, Rule based grammar Checker, Statistical based Grammar Checker, Hybrid Grammar Checker, Subject-Verb, Object-Verb, Adverb-Verb, Noun-Modifier, Word Sequence

Dedication

I am dedicating this thesis to my beloved parents Gebrekiros G/her and Hagosa G/hans, who support and encourage throughout my life. Without your unconditional love and prayers I am not where I am today.

Acknowledgment

First and foremost, I thank my GOD and his mother Saint Mary for letting me live to see this thesis.

I especially thank my advisor, Dr. Yaregal Assabie, for giving me his endless support over the years. I am grateful for his guidance and the opportunities he has afforded me and frequently take me on outings to let me know my work is appreciated.

I would like thank to department of Computer Science for giving me an opportunity to complete this work. I am also grateful to my Instructors, Dr. Mulugeta, Dr. Dida, Dr. Solomon, Dr. Fekade, Dr. Mesfin, Dr. Sebsibie, and Dr. Ayalew for helping me from the beginning till the completion of my study.

I would like to Honor Yemane Ketela for providing me his well-organized and POS annotated Tigrigna Corpus.

I would like to thank my colleague and my friend Kibrom Haftu who is involved in my thesis work in editing my many mistakes. You are truly an outstanding person and an able educator and, I thank you from the bottom of my heart. In addition, I would also like to thank my friends who were involved in my thesis work: Senait Kiros, Mulata Kebede, Hagos G/medhn, Mulu Fisseha, and Axumawit Giday. Without their passionate participation and input, the work could not have been successfully conducted.

I also would like to express my wholehearted thanks to my dearest wife Samrawit Tsegay and my daughter Diana who lead me through the valley of darkness with light of hope and support.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Acronyms and Abbreviations

AVA	Adverb-Verb Agreement
IR	Information Retrieval
MFAE	Morphological Feature Based agreement Extraction
MNA	Noun-Modifier Agreement
NLP	Natural Language Processing
OVA	Object-Verb Agreement
POS	Part of Speech
SOV	Subject-Object-Verb
SVA	Subject-Verb Agreement
QA	Question Answering
WSA	Word-Sequence Agreement

Chapter One: Introduction

1.1 Background

The development of computer technologies has been having a strong impact on the world; often introducing major changes into certain fields of human activities within human communities [1]. To perform those major changes, written languages communication play important roles. Written language communication occurs by means of writing, which besides the traditional paper and pen, is facilitated by the computer, the internet and other applications such as the mobile phone. Word processing and sending messages via email are among the most usual activities on computers [2].

Natural Language processing (NLP) means computer-aided processing of language produced by a human. But, human language is inherently irregular and the most reliable results are obtained when a human is involved in at least some part of the processing. As a field of scientific inquiry, it plays an important role in increasing computers capability to understand natural languages, the language by which most human knowledge is recorded [3]. A lot of applications are performed in the field of NLP to make human language easily understandable by machine or computers such as question answering, grammar checking, spelling checking, etc [4]. A grammar checker determines the syntactical correctness of a sentence. Grammar checking is mostly used in word processors and compilers [5]. Grammar checking for application like compiler is easier to implement because the vocabulary for programming language is finite but for natural language it is challenging because of infinite vocabulary [6]. Grammar checkers check the grammatical structure of sentences based on morphological processing and syntactic processing. These two steps are part of natural language processing intending to understand natural languages [7].

Grammar checker is helpful in a variety of scenarios, such as text authoring and language learning. It uses to find grammatical errors in the input text: incorrect use of person, number, case or gender, improper verb government, wrong word order, and so on [8].

Tigrigna is a typically Semitic language in which the word order in clauses is subject–object–verb usually [9]. For instance, sentence of Tigrigna: (ሃይለ ተምሃራይ እዩ ።/ Haile temiharai eieu (Haile is a student.) Subject = ሃይለ/haile, object = ተምሃራይ/temiharie, verb = እዩ/ieu, punctuation =:: However, Tigrigna is unique among the Semitic languages in several ways such as: Tigrinya

has compound prepositions such as: ኣብልዕሊዓራት/ ab leliarat (on (top of) the bed). Preposition= (ኣብ/ab, Preposition= ልዕሊ/lə‘li and Noun= ዓራት/arat [10].

1.2 Motivation

The advent of personal computers and mobiles has increased communication between Tigrigna speakers using electronic texts such as newspapers, emails, weblogs, books and thesis are produced daily. Grammar checker is very useful when user needs to write some kind of text in that case, user does not need to focus on the grammar errors and producing grammatically error free electronic documents has benefits of easy data organizing and data management. However, the absence of Tigrigna grammar checker leads to many Tigrigna documents that can be seen as having grammatically incorrect sentences. For instance, ኣብ ቀማሊ ወርሒ ናብ መቐለ ክይደ። (I was go to Mekelle next month.) Adverb verb disagreement, (እቶም ቅብሊታት ንጀመርቲ መስተኣናገድቲ ኣድላዩ እዩ። (The receipts is good for beginner waiters.) Subject/verb miss agreement and etc. in such situations, users need an automatic grammar checker/corrector that can help to improve the quality of electronic texts. This has motivated us to develop a grammar checker for Tigrigna text to ensure that what the users’ input is grammatically correct or not.

1.3 Statement of the Problem

A grammar checker has been developed for different languages such as English [2, 6, 11], Swedish [12, 13], Chinese [14], and etc. to detect syntactical text input errors and to make a component of different NLP applications such as question answering, semantic analysis, dialog system, etc. It is now common to see grammar checkers for such languages as components of word processing applications such as Microsoft Word. Grammar checkers have also been developed for Amharic [15] and Afan-Oromo [16]. The Amharic grammar checker was developed using a hybrid of statistical and rule-based approach whereas the Afan-Oromo grammar checker was developed using rule based method. Tigrigna grammar construction is different from Amharic, English and other languages. Due to such differences, each language has a specific set of characteristics which need to be considered in the development of grammar checker. To the best of our knowledge, research and development work on Tigrigna grammar checker has not been attempted so far. It has been indicated that a hybrid approach comprising statistical and rule based technique provided good result for Amharic [15]. Although Amharic and Tigrigna share similar linguistic characteristics, they still have fundamental underlying

differences such as morphological processing, syntax processing, etc. Thus, we can infer that a hybrid approach of statistical and rule based techniques could be applied to develop a grammar checker for Tigrigna by dealing with the unique features of the language.

1.4 Objective

General Objective

The aim of this thesis is to design and develop a grammar checker for the Tigrigna language using hybrid of statistical and rule based approach.

Specific Objectives

In line with achieving the general objective stated above, the research would attempt to address the following specific objectives.

- Review the basic useful properties of Tigrigna language grammar;
- Prepare corpus collection of simple and complex sentences that would potentially serve for the experiment;
- Model the syntactical and morphological rules of Tigrigna sentences;
- Design grammar checker algorithm for Tigrigna text;
- Develop a prototype of Tigrigna Grammar Checker; and
- Evaluate the performance of the developed prototype.

1.5 Methodology

Literature Review

In order to understand the problem we will review books, articles, journals and other publications related to Tigrigna language. Approaches to the development of Grammar checker, Grammar errors, and Evaluation Metrics of Grammar Checker will be explored.

Data Collection

Corpus is one of the resources required in natural language processing researches and a good sized text can reasonably show a language's grammatical behavior. For the purpose of this research, we will use a corpus annotated with POS from [17].

Prototype Development

In this thesis, we will use tools such as C# programming language, Microsoft Visual Studio, Notepad++, SQL, and SharpNlp toolkit.

Evaluation

In order to assess whether the main goal of this study is achieved or not and the degree of achievement, the outcome of the system will be evaluated by using well-known evaluation metrics such as precision, recall, and F-measure.

1.6 Scope and Limitations

The research aims to a develop grammar checker for Tigrigna language that detects grammar errors in Tigrigna written texts. The study has no correction mechanism of the detected errors in the written text.

1.7 Application of Result

The result of this thesis work has many applications. The system is useful for Tigrigna text producers to detect grammatical errors in the text. Tigrigna Grammar checker will be used in many areas of NLP for Tigrigna language such as in Question Answering, Information Retrieval, Next Word Prediction, Text Summarization, Text Categorization, Speech Recognition, Semantic Analysis and etc.

1.8 Thesis Organization

The organizations of the remaining parts of this thesis are described in this section. Chapter Two discusses the literature review. The reviews of related works regarding Grammar checker studies are also presented in Chapter Three. Chapter Four presents the design of Tigrigna Grammar Checker. Chapter Five discusses the implementation and evaluation result of the system. Chapter Six provides the conclusion of the thesis and recommendation.

Chapter Two: Literature Review

2.1 Introduction

In this chapter, topics that will set landscape for the rest of the thesis will be presented in order to get the important terms and concepts that may be used in different sections of the upcoming chapters. The subject of this chapter includes Tigrigna language, approaches to development of grammar checker, and followed by grammar checker evaluation. In the Tigrigna language section, the part of speech (POS), morphology, grammar structure, and grammar errors are discussed in detail. In the approaches to development of grammar checker section, the issues of grammar checker system and the approaches of rule based grammar checker, statistical based grammar checker, syntax based grammar checker, and a hybrid grammar checker are presented respectively. In the remaining section, the lists of grammar checker evaluation metrics and their mathematical calculation is presented.

2.2 Tigrigna Language

Tigrigna belongs to the Semitic family of language. It is the official language of Eritrea and Tigray regional state of Ethiopia [10]. The language is spoken by about 7million people worldwide [9]. It uses Ethiopic script for writing [10]. Tigrigna has become the primary language of oral and written communication in Eritrea and Tigray [10]. The grammar of Tigrigna language regarding POS, morphology, grammar structure, and grammar error is described in this topic.

2.2.1 Tigrigna POS

POS is an appropriate given class of word, in which each word of a written text has a unique word class. Words of Tigrigna language can be classified into eight general classes [19]. These are preposition, noun, verb, adjective, adverb, pronoun, determiners, and conjunctions. A brief description of each of the classes is given in the following topics.

Tigrigna Noun

Tigrigna noun is a word class that is used for labeling thing, an idea (for example, love), an action, a situation, and phenomena. Common Tigrigna noun classes are Name of persons (ብርሃነ/Birane, ሰናይ/Senay, ጀማል/Jemal, ራህዋ/Rahwa, ሳምራዊት/Samrawit... etc), Name of Places (መቐለ/Mekeelle, ዓድዋ/Adwa, ሓውዜን/Hawezien, ማይጨው/Maichew, and Others.), Name of Animals (ላሕሚ/Lahmi (Cow), ደርሆ/Derho(Hen), ፈረስ/Feres (Horse) , and Others), Name of Natures, Facts, situation(ዛሕሊ/Zahli (Cold), ዋዒ/Waei (Hot), and others), Abstracts(tsehay (Sun), /egziabher(God), fiqri(Love) , and Others.), and etc.

Tigrigna Pronouns

A pronoun is a word which is regarded as a subclass of a noun [20]. It can be taken as noun because it can function as a noun and can take the position of a noun. Tigrinya has subject and object pronoun forms. For instance, the words, ኣነ/ane (I), ንሱ/Nisu/ (he), ንሳ/Nisa (She), and etc. are categorized to subjective pronouns and the words ንዓይ/nay (to me), ንዕኡ/neu(to him),ንዓኣ/naa(to her), and etc. also categorized into objective pronouns [9].

Tigrigna Determiners

Determiners specify nouns in terms of their referential status. One of their functions is to supply features of definiteness and indefiniteness to nouns. For example, definite articles express the presupposition that the hearer or the addressee is familiar with or can identify the referent of the noun that is being determined. Based on these grammatical properties, determiners are identified as functional elements, as opposed to lexical elements.

Tigrigna does not have separate class of definite articles and demonstratives [19]. However, there is one form called determiner that serves both functions. For instance, the words like ኣታ/eta (That), ኣቶም/etom (Those), ኣዚ/ezi (This) and etc. are categorized under determiners.

Tigrigna Preposition

Tigrigna employs prepositions to code a wide range of semantic roles [19]. The following prepositional markers are identified in Tigrigna language.

Table 2. 1 Tigrigna prepositions

Preposition	**Used for**
ሳላ/Sala	Purpose, benefit, reason/because of
ኣብ/a^b	Location/at
ካብ/Kab	Source/from
ናብ/Nab	Destination/to
ብ/be	Direction
ብዛዕባ/bizaeba	Definition/topic
ምስ/mis	Partner/ with
ናይ/Nay	Owner/of
ቅድም/Kidm	Before / in front of
ድሕር/dhar	Afterwards

Tigrigna Quantifiers

Tigrigna quantifiers are also classified as functional categories. They specify the referent of a noun in terms of size or amount. They also indicate partitive, existential and universal references. Words like ኩሉ/kulu(All), ገለ/gele(Some), ብዙሕ/bizuh(Many), and etc. are categorized as quantifier.

Tigrigna Verb

A verb is a word that tells us the state of doing or being and possibly the most important part of any text almost in any language [9]. The major subclass of Tigrigna verbs are auxiliary verbs, perfective verbs, imperfective verbs, imperative verbs, gerundive verbs, and relative verbs [17].

Tigrigna Adjective

Adjectives are words that describe or add extra information to a noun. Adjectives in Tigrigna usually precede the nouns that they modify or describe. Words categorized in Adjective are ሐጺር/hatsir(Short), ነዋሕ/newah(Tall), ፀሊም/tselim(Black), ቀያሕ/qeyah(Red), and etc.

Tigrigna Adverb

An adverb is a word that modifies a verb, adjective, sentences or clauses and other adverbs. In many languages adverbs are classified as open classes; however, Tigrigna's adverbs are classified as closed classes [9].

Tigrigna Conjunction

Conjunctions are words that link words, phrases and clauses to create larger grammatical units. Conjunctions were treated as a separate category. However, conjunctions are treated also as a subclass of prepositions. Conjunctions can be distinguished as two categories such as coordinating or subordinating [20].

Tigrigna Punctuation

Tigrinya punctuation marks are two point (፡) standing as a word separator or use instead of single space, (፣) used as comma, (፤) functioning same way as a semicolon, (?) uses as a question mark, and (።) uses as a full stop to end sentences.

2.2.2 Tigrigna Morphology

Morphology is the study of word formation. It tries to discover the rules that govern the formation of words from the smaller meaning bearing units, morphemes, in a language [9]. Morpheme is the building block from which a word is made-up. It could not be broke down further into meaningful parts [9]. Tigrigna has not been studied linguistically especially with respect of computational morphology. In this topic, we describe the morphology of Tigrigna word classes according to their derivation and inflection characteristics.

Tigrigna Noun Derivation: In Tigrigna, nouns can be primary or derived words. For instance the words ከብዲ kebdi 'stomach', and ልቢ lbi 'heart' are primary words; however, the words like ከብደይ kebdäy 'my stomach', ልበይ lbäy 'my heart' are driveved from the primary words by adding postfix + ay/my.

Tigrigna Noun Inflection: Tigrigna nouns inflect for case, number, definiteness, and gender. It has also the nominative case when it is a subject; accusative when it is the object of a verb; and its gender, number, and grammatical case determine genitive when it is the object of a

preposition. Tigrinya nouns have plural, as well as singular forms though the plural is not obligatory when the linguistic or pragmatic context makes the number clear. Most nouns employ the regular (external) plural formation strategy by adding the suffix-at or-tat– depending on whether a noun ends with a consonant. However, some noun plurals may be formed through internal changes ("broken" plural) as well as through the addition of suffixes. For instance, ፈረስ feres 'horse', ኣፍራሰ ’afras 'horses' [10].

In Tigrigna noun, gender is expressed in the pronoun system. Some nouns do not inflect for gender, except for few that are classified. As adjectives or nouns, and thus employ the adjectival gender inflection form, morphemes such as **-ay, -ka, -ki ,-kum , -Kin ,-ka-t-kum** are the most common gender and belongingness indicators in a given noun especially on noun analysis with combination of number and possessiveness. Example: the morpheme /-ay/ suffix on the stem **tamahar_ay** ‘student, its analysis is male in gender, first/second/ third/ person Possessiveness (depends on the pronoun) and singular in number. The morpheme /-it/ is suffixe on tamahar_it “student”, is for gender female. The morpheme /–ka/ is suffixe a on the noun stem /defter/ and its surface output is /defter-ka/ ‘your exercise book’ with male, third person, singular.

The morpheme-**Kin** is suffixe on the stem /geza/ ’house’ and its surface result is /geza-Kn/ ‘your hose’ female, plural and third person possessor. Therefore if any noun stem / verbs and adjectives ends with the morphemes /–ay/, –ka/, - kum, then the noun stem /adjective identifies a gender male [9]. Some of the opposite gender nouns are shown in the table 2.2.

Table 2. 2 Gender based nouns

Gender	
Masculine	Feminine
ብዕራይ/OX	ላሕሚ /COW
ወዲ /BOY	ጋል /Girl
ሓው /Brother	ሓፍቲ /Sister

Tigrigna Verb Derivation: The morphology of the Tigrinya verb is often very complex. Like other Semitic languages, the derivational morphology of Tigrinya is characterized by the root

and-pattern or template morphology. The basic unit of a word is the ‘root’ which constitutes its semantic core. It is an abstract form since it does not exist as a word in the lexicon of a language [21]. In most cases, the root consists of three consonants, also known as radicals, but the number may vary from two to five. These consonants interlock with different vocalic templates and are associated with prefixes and suffixes in order to derive a stem. For instance, the radicals of the root ፀሐፍ/tsehefe combine with the vocalic pattern ¨a to derive the historic perfective stem ፀሓፈ tsehafe ‘he wrote’, and they combine with the vocalic patterns ¨a, i and u, respectively, to derive the simple/gerundive perfective stem ፀሒፉs tsehifu‘he wrote’.

Tigrigna Verb inflection: The verb may be inflected for such things as person, number, gender, aspect, mood, and tense. For example, we use the inflection of the root ፀሐፍ that is taken from [20] shows in the Table 2.3.

Table2. 3 Sample Tigrigna verbs derived from ፅ ሐፍ

Word	Meaning	Gender	Person	Number	Time
ፀሒፈ	I Wrote	-	1st	S	Past
እፀሕፍ	I Write	-	1st	S	Present
ክፅሕፍ	I will Write	-	1st	S	Future
ፀሒፍና	We wrote	-	1st	P	Past
ንፀሕፍ	We write	-	1st	P	Present
ክንፅሕፍ	We will write	-	1st	P	Future
ትፅሕፍ	You write	M	2nd	S	Present
ክትፅሕፍ	You will write	M	2nd	S	Future
ፀሒፍክን	You wrote	F	2nd	P	Past
ትፅሕፉ	You write	F	2nd	P	Present
ክትፅሕፉ	You will write	M	2nd	P	Future
ፀሒፉ	He wrote	M	3rd	S	Past
ይፅሕፍ	He write	M	3rd	S	Present
ክፅሕፍ	He will write	M	3rd	S	Future

Tigrigna Adjective Inflection: In Tigrigna most Adjectives have both gender (feminine and masculine) and number (singular and plural). Like, the morpheme -ቲ/ti marks the feminine gender

along with internal changes in the vocalic templates of some of the consonantal roots table below. Some of the Tigrigna Adjectives are shown in the Table 2.4.

Table2. 4 Tigrigna Adjective Inflections

Adjective	Meaning	Gender	Number
ንፍዕቲ	she clever	Feminine	Singular
ንፋዕ	he clever	Masculine	Singular
ነዋሕ	she tall	Feminine	Singular
ነዊሕ	he tall	Masculine	Singular
ብርትዕቲ	she strong	Feminine	Singular
ብርቱዕ	he strong	Masculine	Singular
ብርቱዓት	They/we/you strong	All	Plural

Tigrigna Pronoun Inflection: The pronoun system reflects exhaustive number, gender and person agreement values. Both subject and object pronoun stems are marked with similar pronominal suffix forms that are distinct for the gender, number and person combination they mark. Some of the Tigrigna pronouns that inflect number, gender and person are listed in Table 2.5.

Table2. 5 Tigrigna Pronouns

Subject Pronoun	Number	Gender	Person
አነ/ane/ (I)	Singular		First
ንሱ/Nisu/ (he)	Singular	masculine	Third
ንሳ/Nisa (She)	Singular	Feminine	Third
ንስኻ/Nisk^a	Singular	Masculine	Second
ንስኺ/Nisk^i	Singular	Feminine	Second
ንሕና/Nihina	Plural		First
ንስኻትኩም/NisK^atikum	Plural	Masculine	Second
ንስኻትክን/NisK^atkin	Plural	Feminine	Second
ንሳቶም/Nisatom	Plural	Masculine	Third

Tigrigna Determiner Inflection: In Tigrigna determiner can inflect the nouns in gender or number. Some of the Tigrigna determiner inflection is shown in Table 2.6.

Table2. 6 Tigrigna determiner inflection

Determiners	Meaning	Gender	Number
እቲ/eti	That	Masculine	Singular
እታ/eta	That	Famine	Singular
እቶም/etom	Those	Masculine	Plural
እተን/eten	Those	Famine	Plural
እዚ/ezi	This	Masculine	Singular
እዛ/eza	That	Famine	Singular
እዞም/ezom	Those	Famine	Plural

Tigrigna Quantifiers Inflection: Tigrigna Quantifiers can inflect nouns in number and gender. Some of these quantifiers that inflect number and gender are listed in the Table 2.7.

Table2. 7 Tigrigna quantifiers Inflection

Quantifier	Meaning	Plural	Singular	Famine	Masculine
ኩሉ	All	✓			
ገለ	Some	✓		✓	✓
ነፍሲወከፍ	Every		✓	✓	✓
ብዙሕ	Many	✓			
ውሑድ	Few		✓		
ሓደ	One		✓		✓
ሓንቲ	One		✓	✓	
ክልተ	Two	✓		✓	✓
ብዙሓት	Many	✓		✓	✓
ውሑዳት	Few	✓		✓	✓
ኩለን	All	✓		✓	
ገሊኣም	Some of them	✓			✓

2.2.3 Tigrigna Grammatical structure

Grammar is often defined as the rule system of a language, typically taught about appropriate usage and about writing [22]. It can be used to refer to a set of rules developed to control certain aspects of the usage of the language, such as in question to create meaning, by building both meaningful words and larger constructions of sentences [23]. Generally, it is a restriction on how words must be arranged to construct a sentence, phrase or paragraph; such restrictions are principles of syntax and semantics. The syntax of a language describes how words can be combined to form sentences, phrases or paragraphs. The semantics of a language regards the modification of words with respect to time number and gender. The grammar syntax and

morphology differ in each language. Conversely, this means that, if there is a difference in syntax or morphology in two languages, these languages are not the same a language can be distinguished from another by its grammar; these languages need to be considered separately [23]. In this section, we describe the phrases and sentences structure of Tigrigna languages.

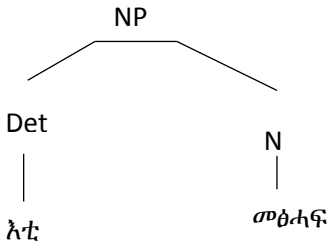
Tigrigna Phrases

The word phrase means group of words that can function as arguments of noun predicates. Every phrase in Tigrigna has head word, in which the head words are in the right position of the phrase [20]. However, in some cases the head word may also written in the left side of the phrase, like: ናብ ዓዲ/nab a^di (to village), in this phrase even the preposition word ናብ/nab is in the left position, it uses as a head; then the phrase is Prepositional Phrase. In this topic we will see the main phrases such as Noun Phrase, verbal phrase, and adjective Phrases in detail.

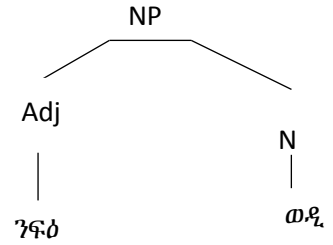
Noun Phrases (NP)

Tigrigna noun phrase consists minimally of a head noun which may be accompanied by optional elements such as quantifiers (Q), determiners (Det) and modifiers (e.g. adjectives, possessives, relatives) [20]. For instance, Noun Phrases are listed here:

1) እቲ መፅሓፍ/ etimets^haf (the book) is NP



2) ንፍዕ ወዲ/ nɪɾue^ wodi (Clever boy).



3) ፃዕዳ ዕምበባ, in this phrase the head ዕምበባ is a noun. Therefore this is called Noun Phrase (NP). In general the structural rule of Tigrigna noun phrase [20] can be formulated as: NP => N//ADJ+N/ADJ+N/PRP+N/NP+N

Verbal Phrase

A verb phrase is composed of a verb as a head and other constituents such as complements, modifiers and specifiers [20]. For instance, መምህር እዩ/memhir eyu(he is a teacher), in this phrase the head word እዩ is a verb class word. Therefore, the text መምህር እዩ is verbal phrase (VP). The general rule of verbal phrase is formed as: VP=>V/N+V/ADJ+V/Adv+V

Adjective Phrase

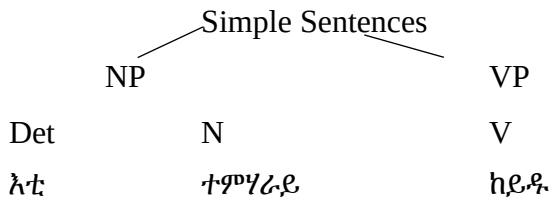
An Adjective phrase is a combination of words in which the head word will be an adjective [20]. For instance, ስጋ ሃራፍ/sega haraf/(meat curios)in this example the head word ends with the adjective ሃራፍ/haraf/ thus, the phrase is an adjective phrase. The general pattern of adjective phrase is: ADJP=>ADJ/N+ADJ/PRE+ADJ/CON+ADJ

Tigrigna Sentences

A sentence is a collection of words expressing a judgment of the mind. A sentence should not be incomplete like a phrase to express something. The general structure of Tigrigna sentences SOV arrangement. The subject can be thought as a noun phrase which is used to mention some objects or persons and the predicate is used to say something true or false about the subject. These two elements of sentences are known as noun phrase (NP) and verb phrase (VP) which are headed as noun and verb respectively. Tigrigna sentences are classified on the basis of numbers of clause and types of clauses present in a Tigrigna sentence means according to the structure, those sentences are divided in to four kinds [20].

Simple sentences

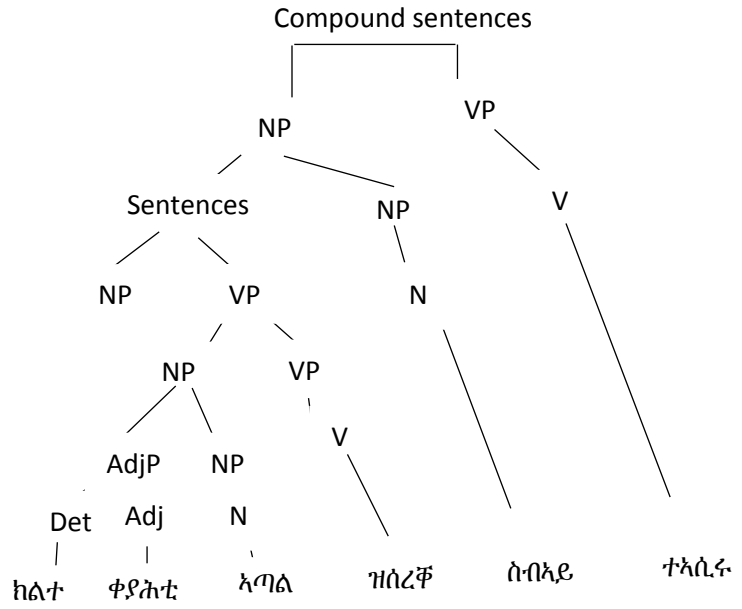
A sentence consists of only one subject and a verb and it expresses complete thought. The pattern of simple sentence is NP+VP. For instance, እቲ ተምሃራይ ከይዱ።።



Compound sentences

A sentence consists of at least two verbs. One verb is at the head of the sentences the others also part of the Verbal phrase or Noun phrase. In this sentences the, subject performs two actions.

Examle: ክልተ ቀያሕቲ ኣጣል ዝሰረቐ ስብኣይ ተኣሲሩ።



Complex sentences

In complex sentences the sentences are constructing from two subjects and two verbs. With these subjects and verbs based on the situation it may also contain determiners and conjunctions.

For instance, in the sentence ኪደነ ነታ ዝፀረፈቶ ተምሃሪት ብበትሪ ወቒዕዋ።, the subjects are ኪደነ and ተምሃሪት, and the verbs are ዝፀረፈቶ and ወቒዕዋ.

Compound Complex Sentences

In these sentences the sentences construct from two subject and three verbs. The first subject performs two actions (verbs), and the second subject performs one action. For instance, in the sentence ንስኻ እንተተፅንዕነይርካ ከም መሓዙትካ እንትሓልፉ ኣይመወደቕካን።, the subject's areንስኻ and መሓዙትካ, and the verbs in this sentence are እንተተፅነንዕነይርካ, ኣይመወደቕካን, and እንትሓልፉ.

2.2.4 Grammar Errors

When people are writing, they make mistakes, mostly the syntactic rules of a language, such as feature agreement, order or choice of constituents in a phrase or sentence, thus concerning a wider context than a single word. Those grammatically error writing are also happen in Tigrigna language users. Grammar error may occur due to the subject's insufficient knowledge of the

language rules or the written language norms that deviate from the already acquired (spoken) grammatical knowledge have to be learned and even can make grammar errors when writing on a computer due to rewriting or rearranging text [23].

Common Tigrigna Grammar Errors

As any other language the grammar of Tigrigna happens the words in a sentence can disagree according to person, gender, cases, tenses and number. Other agreement errors appear between the subject and the verb, which can involve long distance dependencies. In this section we take a look at the most frequent error types in Tigrigna according to Tigrigna Language Experts and Tigrigna text books [20].

Frequent agreement errors in Tigrigna are located within the words in a text can disagree subject and verb, verb and object, adjective and noun, adverb and verb, noun and modifier, word sequence and others in gender, number and persons. Other agreement errors appear between the subject and the predicative, which can involve long distance dependencies.

Subject-Verb Agreement Errors: Is the most common errors with subject-verb agreement are to do with number, person and gender in Tigrigna language. For instance, ምሳሐይ ክበልዕ ኢና።/ misahey kibelie ena/ I will eat our lunch. In this example the subject ኣነ/ane/I which is taken from the object ምሳሐይ and the verb is ኢና/ena/ eat. The singular subject ኣነ/ane is disagreeing with the plural verb ኢና, this make the grammar rule is incorrect. Hence, correct subject-verb agreement has same number that means singular subject takes singular verb. The correct grammar of the sentences is ምሳሐይ ክበልዕ ኢየ::/misahey kebelie eye/I will eat my lunch.

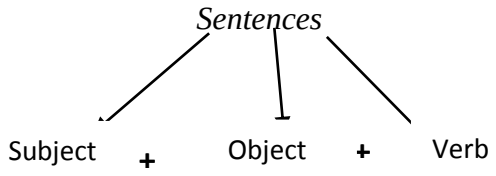
Object -verb agreement error: The same as the subject verb agreement error the object and verb agreement of a sentence is expected to agree in number, gender, and person.

Noun-modifier agreement error: Tigrigna nouns are modified by adjectives, determiners, quantifiers, and others. Adjectives in Tigrigna inflect according to the gender, person and number of the head noun. For instance, in the ነዋሕ ወዲ /newaH wodi/ Tall boy, text the adjective ነዋሕ/newaH (tall) is third person singular feminine; and The noun ወዲ/ wodi (Boy) is third person singular masculine. Both the Adjective and the noun agree in number and person. Whereas, disagree in gender maker. In order to correct the grammatical of this text either the adjective could be changed to male or the noun must be changed to fame gender. Therefore, the correct grammar is either; ነዋሕ ጋል/nwwaH Gual or ነዊሕ ወዲ. The quantifier in noun phrase disagrees

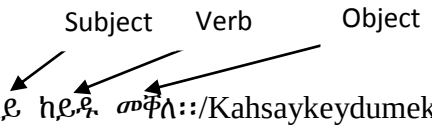
with number agreement. For instance, ሰለስተ ፈረሥ/selsete feres/three horse in this phrase the quantifier ሰለስተ indicates many/plural whereas, the noun ፈረሥ is singular. The number-noun agreement rule in Tigrigna is singular number+ singular noun, or plural number + plural noun, based on this rule the text is grammatically incorrect.

Adverb and Verb Agreement: Tigrigna Adverb usually modifies the first verb that comes next to it in time, place, circumstance etc. The time adverbs describe the time at which an event takes place. These adverbs may show a specific time at which a given action takes place or its duration. Tigrigna verbs indicate the time at which action takes place in relation with the adverbs.

Word sequence agreement error: In Tigrigna grammar rules all verbs go at the end of a statement unlike in English where verbs are placed in the middle of a statement ([Subject] [Verb] [Object]), Tigrigna places its verb at the end ([Subject] [Object] [Verb]) of the text.



Sometimes, in Tigrigna texts occurs error word order like: ([Object] [Verb] [Subject]) or ([Verb] [Object] [Subject]), etc. For instance, in the sentence ካህሳይ ከይዱ መቐለ። has an error in the word order:



ካህሳይ ከይዱ መቐለ።/Kahsaykeydumekele/. In this example the sentence is formed as [Subject, Verb, Object] formation, it considered as grammatically incorrect.

2.3 Approaches to the Development of Grammar Checker

2.3.1 Grammar Checker System

Natural language processing (NLP) can be defined as the automatic (or semi-automatic) processing of human language [25]. It is narrowly used, as it is a large and multidisciplinary field, which includes machine translation, question answering, grammar checker and others for processing human languages. It is often studied to solve the problems of language processing in contrast with part of speech, morphology and other components of the human languages. Based on the linguistic characteristics several NLP studies are performed such as the studies on grammar checker [25, 26]. The purposes of those studies on grammar checker are for examining incorrect texts against the syntax and semantic structure of any language.

A grammar checker system is a computer-based formal system for describing the rules and syntax allowable in the language to detect a construction error. The goal of a grammar checker is to check a sentence or text for its grammatical correctness which helps to prepare an important document. Now, a part of most word processing programs flags what they perceive as stylistic, grammatical, or mechanical problems in a document by highlighting or underlining them, and upon request comment on, explains, and sometimes suggest corrections for each problem [27].

A grammar checker system includes components of tokenization, POST, morphological analysis, and grammar checker [4]. In those systems, ways of checking the grammars of a given sentence is performed by several techniques. However, it includes the tasks such as identifying the words of a given sentence and using a program by capturing and analyzing the generalization of a given word and classifies those words into their POS or other grammatical categories [4, 14, 15].

The tokenization component is used to break input texts from an input file into sentences [16]. The tokenized sentences are further tokenized into words, symbols, or other meaningful elements called tokens. Those tokens are used for grammar checkers to identify the word order and agreement rules in the given text. For instances, the text ትማሊ ቀልጢ ፉመጺኡ።/tmali qelTifu me^Si^u (Yesterday he came quickly.) is tokenized into tokens (‘ትማሊ’, ‘ቀልጢፉ’, ‘መጺኡ’, and ‘።’)[19].

The morphological analyzer component concerns the structure of words [7]. Words are assumed to be made up of morphemes, which are the minimal information carrying unit. Thus, the grammar checker takes morphologically analyzed word gives as input to detect the agreement errors in gender, person, number, time, and etc.

The part of speech tagger component assigns each word and punctuation mark with appropriate parts of speech tag [19]. This component is treated as a main component of grammar checker [16]. Since it gives large amount of information about a word and its neighbor such as word categories like Noun, Verb, Adjective, and others. This is useful in grammar checker system because of words are ambiguous having of more than one possible part of speech and the goal is to find the correct tag for the situation.

The most common grammar checker system development approaches are rule-based, statistical, syntax and hybrid approaches [28].

2.3.2 Rule Based Approach

The rule-based approach is performed by constructing error detection rules manually for the given language. These rules are then used to find errors in the text that has already been analyzed. These rules often contain suggestions on how to correct the error found in the text[31].

According to [16], it consists of sets of rules used to analyze the matching of a text with in specified pattern in which POS tagged is performed.

Generally, rule based approach is used to detects an error of input texts in which the text is considered as having POS tagged by matching against a set of rules previously constructed based on the syntax and morphology grammatical knowledge of the specific language. The grammar checker uses these rules to find errors in each input text by comparing against those rules; if there is any one matched rule the text is considered as correct otherwise the text is grammatically incorrect.

The rule-based approach has some characteristics. It has a strict sense of well-formedness in mind, imposes linguistic constraints to satisfy well formedness, allows the use of heuristics, and relies on hand-constructed rules that are to be acquired from language specialists rather than automatically trained from data[30].

The advantage of this approach is that the grammar checker requires no need of training data, good quality based on language having small amount of electronic data and low coverage of linguistic and it is easy to incorporate domain knowledge into the linguistic knowledge which provides highly accurate results [31].

However, this approach also has some disadvantages. As it is created manually, to build several hundreds of rules it could be time consuming and requires a great knowledge of a specific language.

2.3.3 Statistical Approach

A statistical approach does need a tagged corpus to train the language model (LM) instead of handcrafted grammatical rules. Word sequences which occur often in the corpus can be considered correct in other texts; too, uncommon sequences might be error [3]. In statistical approach, all the required statistical information is extracted from a set of corpora and the text will be compared against the extracted information. The statistics extracted by language modeling techniques are commonly known as n-gram statistics [29]. The problem of this approach that is, performance of a system depends on the amount of the corpus. Specifically, the performance is higher if the amount of the corpus size is large, else low in performance [24].

In general, a statistical approach uses N-gram models in which a language model of the probability of a text, conditioned on some number of previous documents. In this model, the document is viewed as a sequence of n texts and probabilities. The probability of a text in a language is a sub sequence of n neighbored tokens in a sentence, where n defines the number of tokens. The N-Gram probabilities come from a training corpus that is used by the system to detect a sentence which is grammatically incorrect. After the entire text of a document, it is scanned for errors and the result of a scan is a list of all possible errors in the text that measure the probability of the text using n-gram analysis. The possible error examines against the simply setting the threshold high enough for an error should be sufficient to satisfy this requirement. However, a grammar checker with a threshold that is too high will miss a large number of legitimate errors [29].

The fundamental equation of calculating the probability can be written as: P (w/h) where, w is the input text or word, h is the training data or corpus, and P (w/h) is the function of calculating the existence probabilities of w in h. Then the word w is classified as grammatically correct where the probability result meets some threshold otherwise the word is verified as incorrect [25, 3].

2.3.4 Syntax-based Approach

In this approach, a sentence is completely parsed into a tree like structure to check the grammatical correctness. The sentence is said to be correct if the parsing succeeds and incorrect

if parsing does not succeed. The advantage of syntax-based approach grammar checker approach is it is always complete, that means any incorrect sentence will be detected, no matter how unknown the error is. Unfortunately, the grammar checker will only be able to detect that there is some error, but it will not be able to suggest that what exactly the error is. There is also a major problem of syntax based approach; it requires a complete grammar that covers every type of text needed to be checked. There are many grammar checkers but there is no robust grammar checker or a parser available today. Usually, parsers give more than one than result even for correct sentences because they suffer from natural language ambiguity [17].

2.3.5 Hybrid Approach

As its name implies, this approach takes the combination of either rule based approach and statistical based approach or syntax based and rule based by taking the advantages from both approaches to improve the performance of the system [13]. A grammar checker system based on a hybrid approach such as statistical and rule based approaches gives better results than the corresponding uncombined approaches [15].

2.4 Grammar Checker Evaluation

An evaluation of systems result by comparison error and correct text ground truth is common in various NLP systems such as grammar checker, question answering, and etc. Many NLP systems are evaluated using precision, recall, and F-score [31]. Precision describes the proportion of detected correct errors by the system from the given errors. Recall describes the proportion of all detected errors by the system from the given errors. It measure of how much errors the system has detected (coverage of system). F-score is derived from two summary measures such as precision and recall. The F-score is calculated by evenly weighting precision and recall [32].

For instance, there is a *reference* comprising a set of tags representing ground truth. Each tag consists of one or more slots, depending on the tag. (For example, in Named-Entity extraction, each tag has two slots “type” and “extent” while in Template-Element and Scenario-Template extraction. Each system participating in the test produces a response or hypothesis comprising another set of tags, each of which also consists of one or more slots. An algorithm is then used to align the hypothesis against the *reference*. The corresponding slots are then matched and scored as either correct or not. If not correct, the error is marked as either a substitution (incorrect slot), deletion (missing slot), or insertion (spurious slot). The scores are then added up and different measures of performance are computed. It should point out that this paper does not address the

important issues of how to align the hypothesis to the reference or how to decide whether a slot is correct or not. This paper is only concerned with how to compute system performance, once the alignment is completed and the correct/incorrect decisions for all the slots have been made. To help in the analysis that follows, define the following symbols [33]:

N= total number of slots in the reference

M = total number of slots in the hypothesis

C= number of correct slots – those slots in the hypothesis that align with slots in the reference and are scored as correct

S= number of substitutions (incorrect slots) slots in the hypothesis that align with slots in the reference and are scored as incorrect

D = number of deletions (missing slots or false rejections) slots in the reference that do not align with any slots in the hypothesis

I= number of insertions (spurious slots or false acceptances) slots in the hypothesis that do not align with any slots in the reference.

From the above definitions that

$$\mathrm{N} = \mathrm{C} + \mathrm{S} + \mathrm{D} \quad (1)$$

$$\mathrm{M} = \mathrm{C} + \mathrm{S} + \mathrm{I} \quad (2)$$

The total number of correct, substitution, and deleted slots is equal to the total number of slots in the *reference*, N, which is fixed for a given test set. The value of M, however, is in general different for each system being tested. M may be larger or smaller than N, depending on whether insertions are more or less than deletions.

Precision and recall are then defined by:

$$\mathrm{Precission(P)} = \frac{C}{M} = \frac{C}{C+S+I} \quad (3)$$

$$\mathrm{Recall(R)} = \frac{C}{N} = \frac{C}{C+S+D} \quad (4)$$

Precision is the percentage of slots in the hypothesis that are correct, while recall is the percentage of *reference* slots for which the hypothesis is correct. Precision takes account of

substitution and insertion errors while recall takes account of substitution and deletion errors. It is possible to "cheat" at both these precision and recall, in that one can design a system with certain behaviors that are of very limited use but achieve high scores. For recall, if a system simply returns everything possible, then by definition is complete it has returned every correct answer, there are no false negatives. Such a system would have recall of 1.0, but contribute little. If one correct answer can be found, then that is all that needs to be returned. Because no false positives are generated, and the numerator is above zero, so this gives maximum precision 1.0. Together, these metrics can be balanced out. These extreme situations used to exploit them contrast with each other: returning everything gives only a baseline precision, and returning just one thing typically gives a very low recall measure. Therefore, we evaluating using F-score, focus are given to true positives, false positives and false negatives.

F-score is derived from two summary measures Precision and recall and defined as:

$$\mathrm{F-Score} = \frac{2 * P * R}{P + R} \qquad (5)$$

Chapter Three: Related Work

3.1 Introduction

A lot of studies have been done in grammar checker as a task of Natural language processing (NLP) using different approaches for various languages. In this chapter, we discuss earlier grammar checker works conducted to Ethiopian languages such as Amharic and Afaan-Oromo, and non-Ethiopian languages such as English, Swedish, Bangla, Nepali, and Punjabi. In addition, this section also reviews grammar checker works using different grammar checker development approaches such as rule based, statistical, and hybrid.

3.2 Development of Grammar Checker for Ethiopian Languages

Aynadis Temesgen and Yarega Assabie [15] have conducted a research which proposed to identify and detect grammar errors for Amharic text sentences using two different approaches of grammar checker system development, which are statistical and rule based approach. The rule based Amharic grammar checker aimed to identify errors in simple sentences using hand-written grammar rule and consists of three modules such as Sentence Splitter, Morphological Analyzer, and Grammar Relation finder. The functionality of those modules is the sentence splitter module that is used to split input text into sentences and the sentence into words. The morphological analyzer module is performed to assign linguistic meanings to each word. The grammatical relation finder module is performed to assign grammatical relation between words in a sentence, and the grammar rule checker module is used to find a match grammar relation of patterns extracted in the grammatical relation finder module against the hand-written rules. The Statistical Amharic Grammar Checker (STAMGRAM) is used to identify grammar errors in simple and complex Amharic sentences using N-gram probability result of Trigram and Bigram. The performance of both the rule-based Amharic grammar checker and STAMGRAM approaches is evaluated using self-made corpus that contains grammatical errors. Based on the results that the researchers got, they concluded the rule-based Amharic grammar checker has performed 92.45% of precision, and 94.23% of recall. The STAMGRAM performs a precision of 59.72%, and recall 82.69% in Bigram and a precision of 67.14%, and recall of 90.38% Trigram. The researcher stated that the system shows a false alarm due to incomplete grammatical rules and quality of the corpus and recommends that the combination of the rule-based and statistical approach by

mixing the good qualities of the two approaches to detect high level grammatical errors can increase the performance of a grammar checker system.

Debela Tesfaye [16] has conducted a rule-based grammar checker for Afaan-Oromo language to determine grammar error in Afaan-Oromo texts by finding match grammatical rules designed manually. The proposed system in this study has five modules, namely, Tokenization, POST, stemmer, grammar relation finder and suggestion creation modules. The performance of this system is tested against manually prepared real world Afan-Oromo texts based on recall and precision, and the result is 80% and 88.89% respectively. The researcher stated that some correct texts are identified as incorrect, and incorrect texts as correct. Thus, false flags are due to compound, complex and compound complex sentences because most of the grammar rules are constructed for simple sentences.

3.3 Development of Grammar Checker for non-Ethiopian languages

Xing and Wang [11] adapt a hybrid of rule based and statistical approaches of grammar checker system to detect English error texts. The study was developed to find and correct error types of article or determiner, noun number, preposition, verb form and subject-verb agreement errors. In this study, the researches constructs manually rules and develop language model using N-gram based on the Google corpus to detect the grammatical errors in English. The system has five components such as Vform, SVA, ArtOrDet, Nn, and Prep. In the ArtOrDet module the system detects error regarding to article or determiner by select the best candidate from a confusion set of possible article choices {a, the, an, ø}. In the Nn module the system detect the noun errors in the text whether it is singular or plural. In the Prep module, the system corrects system corrects errors for the five most frequently prepositions which are in, for, to, of and on. In the Vform module the system determines and corrects the correct form of a verb in the English texts. In the SVA module the system detects subject-verb agreement errors. The performance of the system based on the precession, Recall, and F-Score is 0.3712, 0.2366, and 0.2890 respectively.

Alam and UzZaman [6] developed a statistical Grammar Checking System for Bangla and English language. The study is proposed by considering the n-gram based analysis of words and POS tags to decide whether the sentence is grammatically correct or not. The proposed grammar checker works first by assign tag for each word of a sentence, next uses n-gram analysis (LM) to determine the probability of the tag sequence. Then if the probability is above some threshold

then the sentence is considered grammatically correct otherwise incorrect. The performance of this system is tested using manually and automatically tagged corpus. Using manually tagged English corpus, the performance of the grammar checker for English is 63% (detected 545 sentences as correct, out of 866 correct sentences). Using manually tagged Bangla corpus, the grammar checker's performance is 53.7% (detected 203 sentences out of 378 correct sentences). The performance of the grammar checker also tested using 34 automatically tagged sentences for Bangla. Then, the grammar checker produced result with lower performance which is 38% correct result. The researchers described limitation in their approach, with possible solutions of future work the grammar checker can be a hybrid system combining both statistical and rule based approach to increase the performance.

Rule based grammar checker for Punjabi language system [28] identifies grammatical errors in Punjabi texts such as modifier and noun agreement, subject and verb agreement, noun and adjective, order of modifier of noun in a noun phrase, order of verb in a verb phrase and the like. To detect the errors the system passes through few steps or phases initially, pre-processing task is done on the input text which is tokenization, morphological analysis, rule-based part of speech tagging, chucking and finally, using the grammatical error checking rule, grammatical errors internal to the phrases and the sentences will be identified and correction suggested. The evaluation of the grammar checker shows precision of 76.79%, recall of 87.08%, and F-measure of 81.61%. The researchers stated that the system generated some false alarms for complex and compound sentences. To reduce these false alarms they indicated to work in the future by combining other approaches.

Domeij and Knutsson [12] investigated the achievement of carefully constructed grammatically error detection rules for Swedish texts combining of probabilistic and rule-based techniques. The error detection rules are optimized using statistics of part of speech bigrams and words in a way that each rule needs to be checked as seldom as possible. The researchers proposed components to detect the grammatically incorrect Swedish texts such as Tagging and lexicon, Error rules, Help rules and erroneously split compounds components. The authors are still working with optimizing the system and improving the error rules. The preliminary test of the system is conducted with the error rules show that a recall rate above 50 percent and a precision rate above 90 percent for agreement errors and erroneously split compounds. In addition the authors stated

that, they compare with other related studies and they got better results than systems that use rule-based.

Bal and Shrestha [25] describes the architectural and system design of the Nepali Grammar Checker, which is in due course of research and development. The development uses a rule based approach with the Grammar Checker consisting of independent modules. These modules then in turn serve as a pipeline for the overall integrated system. The Grammar Checker aims to check the grammatical errors such as nominal and verbal agreement, parts of speech inflections, phrase and clause structure and the different categories of sentence patterns for Nepali. Out of the modules proposed in the architectural and system design of the Nepali Grammar Checker, the research and development of the first two modules, namely the tokenizer and the morphological analyzer has been partially completed. A prototype of the two modules is also ready and in the process of being tested and added complexity hand ling features. As mentioned earlier, the proposed architectural and system design of the Nepali Grammar Checker is subjective to changes as further findings of the research work come out in future.

3.4 Summary

From the related work, we have reviewed researches done for Ethiopian Languages and non-Ethiopian languages using different approaches. We understand that the rule based grammar checker systems have drawbacks such as having false flags to detect texts such as compound, complex, and compound complex sentences. The statistical grammar checker approach system also requires large size grammatically correct and POST annotated corpora. The hybrid of statistical and rule based grammar checker approach shows better performance by compromising the drawbacks of each approach. Even if new hybrid systems are available, they didn't identify and detect Tigrigna grammar errors because the techniques are language dependent and the way they are being applied depends on the characteristics of the language. On the other hand, to our best knowledge, there is no grammar checker developed to detect grammatical errors in Tigrigna text.

Thus, in this work, we aim to develop Tigrigna grammar checker. We hypothesize that a hybrid of rule based and statistical approach provides better results for identification of the grammar errors in Tigrigna texts.

Chapter Four: Design of Tigrigna Grammar Checker

4.1 Introduction

In this chapter, we design a Hybrid Tigrigna grammar checker system that detects grammatical errors in Tigrigna senesce using manually prepared grammatical rules and tag sequence and the results analyzed from a single corpus. The system architecture and detail description of modules and components in the modules also follows.

4.2 System Architecture

In this section, we explain our system architecture by specifying list of particular components' function with specific approach to perform its task in general. Specifically, we proposed a Tigrigna grammar checker to detect grammatically incorrect written Tigrigna texts with a hybrid of rule based and statistical based approach. Our system consists of five main modules such as tag sequence gathering, preprocessing, rule based grammar checking, statistical grammar checking, and grammar error filtering. To describe the functionality of those modules, first, the tag sequence gathering module performs a unique tag sequence probability to fill the language model with a list of tag sequence and their probability. Second, the preprocessing module is present to tag tokens within a morphologically based POS tag. Third, the rule based grammar checker module is responsible to detect agreement grammatical errors of the tagged sentence. Fourth, the statistical grammar checker module is used to detect word sequence agreement grammatical errors. Finally, we present grammar error filtering module which is responsible to display grammar error messages. The overall Architecture of the system is be seen in Figure 4.1.

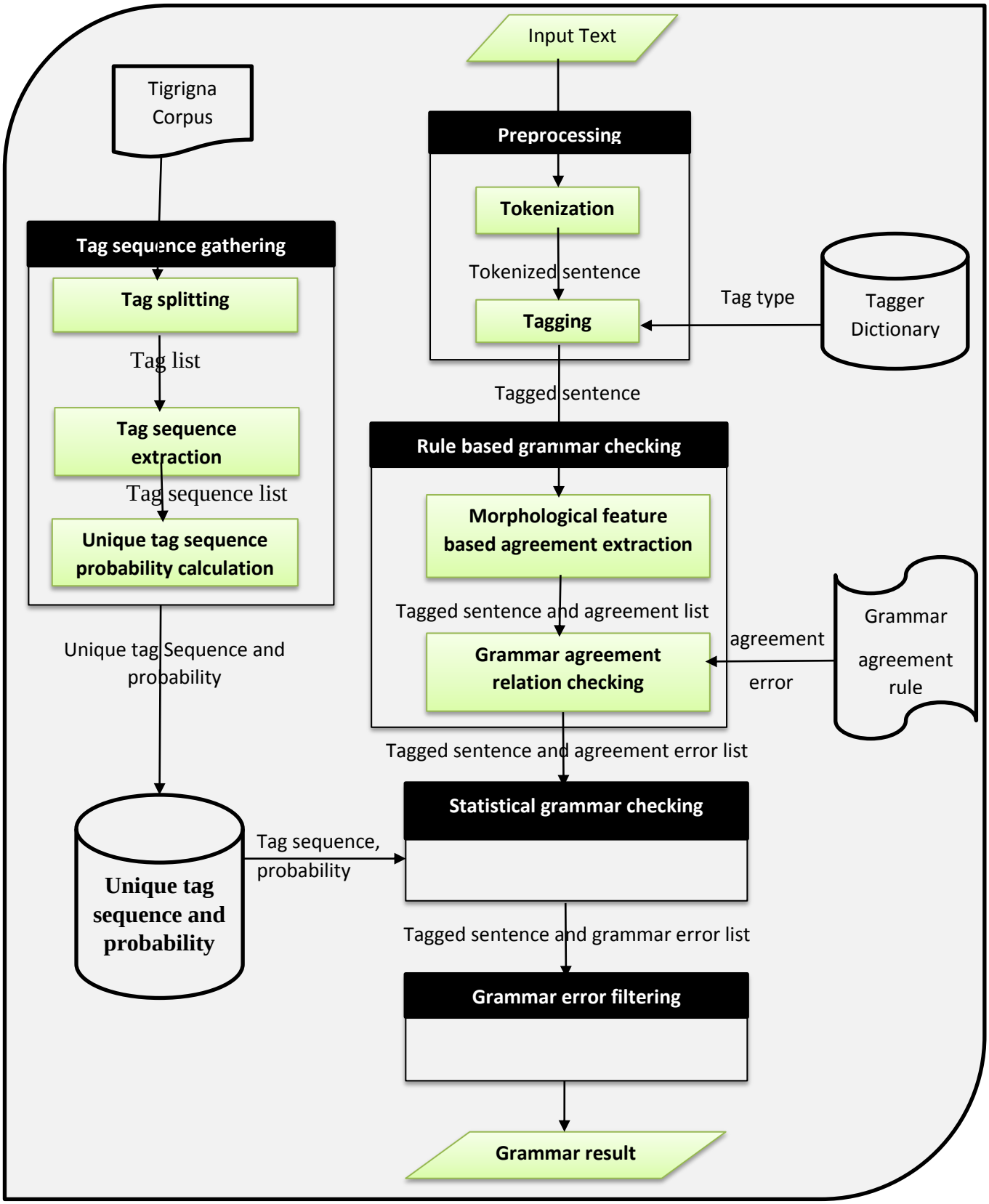


Figure 4. 1 Hybrid Tigrigna Grammar Checker System Architecture

4.3 Tag Sequence Gathering

In this module, tasks are performed to extract the tag list from Tigrigna corpus and it calculated the probability of unique tag sequence from the list. Specifically, this module has three main components namely tag splitting, tag sequence extraction, and tag sequence probability calculation. The tag splitter component splits tags from the tagged corpus; the tag sequence extraction extracts list of tagged sequences. The unique tag sequence and their probability is calculated and filled in to the language model by the unique tag sequence probability calculation component.

Our corpus has Tigrigna sentence which is structurally annotated by morphology and part of speech (POS). Each sentence starts with the <s> and ends </s> token, each word in the sentence is tagged inside the <w> and </w> markers and the punctuations in the sentence are inside the <c> and </c> markers as shown in Figure 4.2.

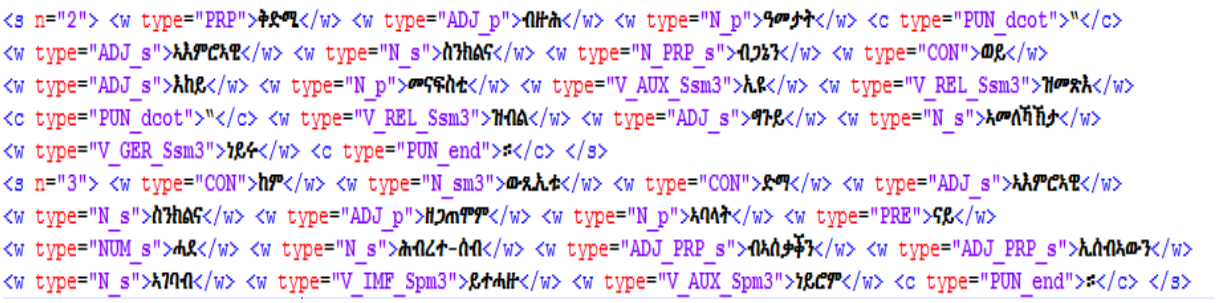


Figure4. 2 Sample sentence in the Tigrigna corpus

4.3.1 Tag Splitting

The tag splitting component is responsible to split the corpus into a list of tags and returns the tag list in an array list structure. It starts by taking the corpus as input, and then it split the corpus into a sentence and then each sentence into tag list. Each tagged sentences in the corpus were splitted based on the sentence start <s> and sentence end </s> tokens; and it split each sentence into tag list by adding <start> and <end> tags to the beginning and the end of the sentence respectively. Each tag of the tokens in the sentence are also separated for word tokens by <w></w> and for punctuation tokens by<c></c>.

For instance, to split the sentences of the above corpus as seen in Figure 4.2. To tag the list, the tag splitting component performs tasks such as, first find the starter of the sentence (<s>) in the

corpus, then identify each token until it gets end sentence (</s>), then it groups all token array in the sentence level as shown in the Table 4.1.

Table4. 1Tigrigna sample Tagged sentence array list

Index	***Sentence List***
0	<start><w type="PRP">ቅድሚ</w><w type="ADJ_p">ብዙሕ</w><w type="N_p">ዓመታት</w><c type="PUN_dcot">“</c><w type="ADJ_s">ኣእምሮኣዊ</w><w type="N_s">ስንክልና</w><w type="N_PRP_s">ብጋኔን</w><w type="CON">ወይ</w><w type="ADJ_s">እከይ</w><w type="N_p">መናፍስቲ</w><w type="VAUX_Ssm3">ኢዩ</w><w type="V_REL_Ssm3">ዝመጽእ</w><c type="PUN_dcot">“</c><w type="V_REL_Sm3s">ዝበል</w><w type="ADJ_s">ግጉይ</w><w type="N_s">ኣመለኻኽታ</w><w type="V_GER_Ssm3">ነይሩ</w><c type="PUN_end">።</c><end>
1	<start><w type="CON">ከም</w><w type="N_s">ውጺኢቱ</w><w type="CON">ድማ</w><w type="ADJ_s">ኣእምሮኣዊ</w><w type="N_s">ስንክልና</w><w type="ADJ">ዘጋጠሞም</w><w type="N_p">ኣባላት</w><w type="PRE">ናይ</w><w type="NUM_s">ሓደ</w><w type="N_s">ሕብረተ-ሰብ</w><w type="ADJ_PRP_s">ብኣሰቃቕኑ</w><w type="ADJ_PRP_s">ኢሰብኣውን</w><w type="N_s">ኣገባብ</w><w type=" V_IMF_Spm3">ይተሓዙ</w><w type=" V_AUX_Spm3">ነይሮም</w><c type="PUN_end">።</c><end>

Second, from the group of array list of tokens in the sentence level needs to identify each tokens tag type using <w type=”tag”> </w> for a words and for punctuation <c type=”tag”> </c>. Some of the array tag list of the spllited tagged sentence array in the Table 4.1 is shown in the Table 4.2.

Table4. 2Tigrigna sample Tag array List

Index	**Tag List**
0	<start>
1	PRE
2	ADJ_p
3	N_p
4	PUN_dcot
5	ADJ
6	N_s
7	N_s
8	CON
9	ADJ_s

In general, the overall task done by the tag splitting is expressed in Algorithm 4.1.

```
Read Corpus
Set Sentence_List[]=corpus split by <"‹s> </s">,
Set Tag_List[]=""
For each sentence in Sentence_List
     Set Token_List[]=sentence split by("<w> </w>" or"<c> </c>")
     Add <sart> in Tag_List
     For each token in Token_List
          Add tag_of token in Tag_List
     EndFor
     Add <end> in Tag_List
EndFor
Return Tag List
```

Algorithm4. 1Tigrigna tag splitting

4.3.2 Tag Sequence Extraction

This component is used to extract sequence of tag list array from the tag list which gets from tag splitting component. It starts by taking of the array tag list as input, then it extracts a sequence of tag list with the range of <start> and <end> tags; in which the <start> and <end> indicate that tag sequence in an individual sentence prepared by 3-tag sequence and 2-tag sequence of lists.

Using the 3-tag sequence we extract a group of three tags which occurs sequentially within a sentence. First, it starts by taking the tag list array, then extracts each 3-tag sequence in the tag list starting from the <start> into the <end> tag, when it reaches the <end> marker it starts from the next tag which is <start> maker of the next sentence. In each tag sequence of a sentence consists a list of three tags which are the tag $_{i-1}$ is the previous tag, tag $_{i}$ is the current tag, and tag $_{i+1}$ is the next tag in the tag list. For instance, the array tag list in Table 4.2 is extracted to 3-tag sequence array list as shown in the Table 4.3.

Table4. 3 Tigrigna sample 3-tag sequence array List

Index	**3-tag sequence**
0	<start> , PRP , ADJ_p
1	PRP , ADJ_p, N_p
2	ADJ_p, N_p, PUN_dcot
3	N_p, PUN_dcot, ADJ
4	PUN_dcot, ADJ, N
5	ADJ, N, N_s3
6	N, N_s3, CON
7	N_s3, CON, ADJ_s

Using 2-tag sequence it extracts a sequence of two tags from the list of tag lists. For instance, the array tag list in Table 4.2 is extracted to 2-tag sequence array list as shown in the Table 4.4. The task of the tag sequence extracted is expressed in Algorithm 4.2.

Table4. 4 Tigrigna sample 2-tag sequence List

Index	**2-tag sequence**
0	<start> , PRP
1	PRP , ADJ
2	ADJ_p, N_p
3	N_p, PUN_dcot
4	PUN_dcot, ADJ
5	ADJ, N
6	N, N_s3
7	N_s3, CON

```
Read Tag_List
Set threetagseq_List[]="", twotagseq_List[]=""
Set tag1= 0, tag2= 0, tag3= 0
For i=0; i< Tag_List;i++
     tag1= Tag_List[i], tag2= Tag_List[i+1],tag3= Tag_List[i+2]
     If tag1 or tag2 is not <end>
          Add tag1, tag2, tag3 in Threetagseq_list
     End IF
END For
Set tag1= 0, tag2= 0;
For i=0; i< Tag_List;i++
     tag1= Tag_List[i], tag2= Tag_List[i+1]
     If tag1 is not <end>
          Add tag1, tag2 in Twotagseq_list
     End IF
END For
Return threetagseq_List, twotagseq_List
```

Algorithm4. 2 Tigrigna tag sequence extraction

4.3.3 Unique Tag Sequence Probability Calculation

This component is responsible to calculate the probability of each unique tag sequence list that comes from the Tag Sequence Extraction component by using the N-gram statistical language model in order to calculate the probability of the tag sequence. Specifically, N is given as N=2 and N=3 applied to calculate the probability of 2-tag sequence and 3-tag sequence respectively and then the tag list and its probability is recorded in the data base of Trigram and Bigram tables.

```
Read Threetagseq_list,Twotagseq_List,Bigram, Trigram
Set CountTagseq=0, tagseqrobablity=0, Count_tag=0
Set total_tagseq=Threetagseq_list.Count + Trigram.Count,
For tag in Threetag_List
     If tag not in Trigram
           Count_tag=count tag in Threetagseq_list
          tagseqrobablity= Count_tag/ total_tagseq
          Add tag, tagseqrobablity in Trigram
      End If
End For
SET CountTagseq=0, tagseqrobablity=0
Set total_tagseq=Twotagseq_List.Count + Bigram.Count
For Tag in Twotagseq_List
     If tag not in Bigram
         Count_tag=count tag in Twotagseq_List
         tagseqrobablity= Count_tag/ total_tagseq
          Add tag, tagseqrobablity in Bigram
      End If
End For
```

Algorithm4. 3Tigrigna tag sequence probability calculator

Generally, the 2-tag and 3-tag sequences and their probabilities are used by the statistical grammar checker to detect word sequence grammar errors in a sentence. We choose lower sequence of tag which is 2-tag sequence and 3-tag sequence instead of higher sequence of tag

such as 4-tag sequence or 5-tag sequence. Since most of the time people write lower and different word sequences, higher sequence of tags may lead to some zero conditional probabilities. For this reason, we use the 2-tag and 3-tag sequence for our system to determine whether the word sequence of input sentence is correct or not.

4.4 Preprocessing

This module has two main components namely tokenization and tagging components. The functionality of the tokenization component is tokenizing Tigrigna input texts into tokens with in a sentence level and also tagging component is responsible for returning the tag type of tokens with the help of tagger dictionary.

4.4.1Tokenization

As mentioned above, this component is responsible for performing splitting of input text as a sentence level; if it is a sentence or phrase into tokens. However, if it is a paragraph, it split into sentence then the sentence into tokens. To split the input text into, a sentence it uses punctuations like full stop (double colon (::)), question mark (?), and exclamation(!) then finally gets tokens within a sentence which are separated by white space or the single colon punctuation (:).

To accomplish its specific task, it starts by reading the Tigrigna input text, then it splits the text into separated sentences, then each sentence into tokens, and it adds <start> and <end> sentence markers. For instance, the system splits the input text **አበበ ናብ ቤትትምርቲ ከይዳ** into seven tokens, and it adds the <start> and <end> marker to the sentence. Finally, it returns the token list፡ **‘<start>’, ‘አበበ’,’ናብ’,’ቤትትምህርቲ’,’ከይዳ’,’::’,’<end>’.**

4.4.2 Tagging

The tagging component is also used to tag tokens within their appropriate morphology based POS tag from the tagger dictionary. It starts by accepting token list in a sentence from the tokenization component as input and assigns each token with a tag by the help of tagger dictionary. The tag has information ơf tokens in word class (POS) and its inflection information (morphology) of each word class. The tokens which are not in the tagger dictionary are tagged as “Unknown”. Table 4.5 shows sample tag type in the dictionary and their descriptions.

Table4. 5 sample tag type in the tagger dictionary

Tag type	**Description**
N_s	Noun singular in number
N_sm3	Noun singular in number, masculine in gender, and third person
N_sf	Noun singular in number, and feminine in gender
N_s1	Noun singular in number, first person
PRO_p	Pronoun plural in number
V_GER_Ssm_Of	Gerundive verb, Subject(singular in number , masculine gender), and Object feminine
NUM_P	Plural number (plural Quantifier)
V_PRF_Ssm3	Perfective Verb, and Subject (singular in number , masculine gender, and third person)
ADJ_s	Adjective singular number
ADJ_p	Adjective plural number
N_pf	Noun plural number, and feminine gender
V_IMV_Ssm3_Os1	Imperfective Verb, Subject(singular, Masculine, and Third person), and Object (singular, and first person)

For instance, the tagging component in this system returns tagged sentence for the tokenized input in the sentence**(‘<start>’, ‘አበበ’,’ናብ’,’ቤትትምህርቲ’,’ከይዳ’,’::’,’<end>’).**as: first, the tokens <start> and <end> are considered as the starting and ending marker tokens of a sentence and they are not tagged. The system simply adds these tokens into the tagged sentence. Second, the token አበበ is noun singular number and masculine gender tagged in the tagger dictionary as N_sm (Noun Singular Male) is tagged as አበበ| N_sm and added into the tagged sentence; similarly the other tokens are tagged. Finally, the tagged sentence of the tokenized sentence that can be returned is **(<start>, አበበ| N_sm, ናብ|Det, ቤትትምህርቲ|N_s, ከይዳ|V_AUX_Ssf3, ::|PUN_End,<end>).**

The overall process and task of the preprocessing module is shown in the Algorithm 4.4.

```
Read InputText,TaggerDictionary
Split InputText into Sentence// based on sentence end punctuation
For Each Si in Sentence //where Si,i=1,2,3,…,n, n is number of sentence
     add <start> to Tokenized_sentence// sentence start maker
      split Si into Token// tokenized sentence si using white space or double colon (:)
      For each Ti in Token //Where Ti,i=1,2,3,…,n n is number of sentence
     Add Ti to Tokenized_sentence
     End for//
     add <end> to Tokenized_sentence// sentence end maker
End For// end of tokenization
Set tagged_sentence is empty
For Ti in Tokenized_Sentence
     If Ti is <start> or <end>
          Add Ti in tagged_sentence
     Else If Ti is in TaggerDictionary
          Add Ti/tag of Ti in   tagged_sentence
     Else
          Add Ti/Unknown in   tagged_sentence
End for
Return tagged_sentence
```

Algorithm4. 4 Preprocessing module

4.5 Rule-based Grammar Checking

This module is used to identify grammar errors regarding Tigrigna language, by specifying the grammar errors in agreement with grammar error type. The agreement error includes Subject-Verb, Object-Verb, Modifier-Noun, and Adverb-Verb grammar agreement errors. To determine these agreement error types, the system uses a set of Tigrigna grammar rules which can be manually designed. In order to perform the determining of grammar agreement error, this module has two components, namely morphological feature based agreement extraction and grammar agreement relation checker. The Morphological feature based agreement extraction is used to extract the grammar agreement information in the tagged sentence of the input text. The Grammar agreement relation checking is applied to determine whether the extracted agreement information is correct or not according to the rule.

4.5.1 Morphological Feature Based Agreement Extraction

The Morphological feature based agreement extraction (MFAE) component is used to extract error types within Noun-Modifier agreement, Subject –Verb agreement, Object-Verb agreement in gender, person, and number, and adverb verb agreement in time, place and manner. To do this the system uses tag pattern information of the tag list in the sentence.

To etxtract Noun Modifier agreement it uses Noun (N) head tag and Modifiers of determiner (Det) or Adjective (Adj) or Pronoun (PRO) or Number (Num) pattern. To perform this we develop an algorithm which is responsible first to look up a head tag pattern noun (N) with a modifiers Adj, PRO, NUM, etc. ; then it being labeled the pattern of the noun and the modifier before it as a Noun Modifier Agreement.

For instance, the Noun Modifier Agreement in a tagged sentence **(እቶም/PRO_pm3, ተምሃሮ/N_p, ከይዶም/Vp_Spm3, ::/PUN_End)** is extracted as follow: First, the system reads tagged (እቶም/PRO_pm3, ተምሃሮ/N_p, ከይዶም/V_AUX_Spm3, ::/PUN_End). Second, it looks a head noun tag with preceding modifier patterns then it extracts the agreement from both the noun and the modifier. In this example, the agreement from the head nouns (ተምሃሮ/N_p) and modifiers **(እቶም/PRO_pm3)** that modify the noun can be extracted as Noun Modifier Agreement.Finally, the agreement is marked for the noun ተምሃሮ/N_p and the pronoun modifier እቶም/PRO_pm3 in which the noun is plural in number and empty in gender and person; the modifier is plural in number, masculine in gender, and third in person. The Noun Number Agreement Extraction

assigns the tag pattern of the noun **N_p** associated with the modifier **PRO_pm3** agreement in to Agrrement_in_sentence = N_p+PRO_p. The work flow of Noun Modifier Agreement Extraction is shown in Algorithm4.5.

```
Read tagged_sentence
For i=1; i<tagged_sentence.count; i++
     N= tagged_sentence [i]
     If N is noun
          for m=i-1; tagged_sentence[m]is Adjective or Determiner or
          Number  or Pronoun;m--
              M= tagged_sentence[i-1]
              Add N of tag + M of tag into Agreement_in_sentence
          End for
     End If
End For
Return Agreement_in_sentence
```

Algorithm4. 5 Tigrigna Noun Modifier Agreement extractions

The extraction of Adverb-Verb agreement is similar to Noun Modifier agreement extraction which differs to extract adverb and verb agreement in time, place, and manner. To perform this, the first adverb modifies the first verb in the sentence. Accordingly, the extraction of adverb-verb agreement is shown in Algorithm 4.6.

```
Read tagged_sentence
For i=0; i<tagged_sentence.count; i++
     Adv= tagged_sentence [i]
     If Adv is Adverb
       For j=i+1; j<tagged_sentence.count;j++
          Ver= tagged_sentence[j]
          If Ver is verb
              Add Adv of Time + Ver of Time, Adv of place + Ver
     ofPlace,AdvofManner+VerofManner in  Agreement_in_sentence
          End If
        End for
       Set i=j+1
     End If
End For
Return Agreement_in_sentence
```

Algorithm4. 6 Tigrigna Adverb verb agreement extractions

The subject verb agreement between the subject and verb in a sentence agreement is done according to number, gender and person tag value of the subject and the verb. The subject of a sentence taken from the tagged sentence in which it tagged as noun (N) or Pronoun (PRO) and the verb is taken from the tag pattern Verb (V). The first noun or pronoun and verb tag in the sentences is considered as the subject-verb agreement; this is when the sentence consists of one subject and one verb; but may be a sentence can consist of more than one subject and verb. So, the first subject is associated to the first verb and the next subject which comes after the verb is also considered as the second subject and if there is a verb after it will be taken as the second subject verb agreement. It continues finding consecutive noun and pronoun in the sentence, it means if the sentence is complex sentence or complex compound sentence it may have more than two subject and verbs.

Each subject verb agreement will consist of three parts such as number, person, and gender for both subject and verb. For instance, (*ኦቶም/PRO_pm3, ተምሃሮ/N_p,* ናብ/PRE, ትምህርትቤት/N,

መጻሉ/Vp_Ssm3_O_ ::/EPUN). In this sentence the first pronoun word in the pattern እቶም/PRO_pm3 indicates the subject of the sentence and the verb word in the pattern መጻሉ/Vp_Ssm3_O indicates the verb of the sentences. The agreement will be taken from the pattern of the subject PRO_pm3 and the verb subject marker Vp_Spm3, and the Grammar Agreement Making creates the Subject verb agreement as: SVA (subject=*እቶም*, *verb*=*መጻሉ*):Sp+Vs, Sm+Vm, S3+V3.Agreement_In_sentence= Sp+Vs, Sm+Vm, S3+V3

The following is our proposed algorithm for Subject Verb Agreement Extraction:

```
Read tagged_sentence
SET S="", V=""
For k=0; k< tagged_sentence.count;k++
     If (tagged_sentence[k] is Noun or Pronoun)
           For v=k+1; v< tagged_sentence.count; v++
               If (tagged_sentence[v] is Verb)
                    S= tagged_sentence[k]
                    V= tagged_sentence[v]
     Add(S of number+ V of number, S of gender+ V of gender, S of
person+ V of person) in Agreement_in_sentence
               End If
           SET k=++v
          END For
      END If
END For
```

Algorithm4. 7 Subject verb agreement extractions

The same as the subject verb agreement the object and verb agreement of the sentences will be created. The object of a sentence will be taken from the tagged word comes as a second noun (N) or Pronoun (PRO); and the verb of the sentence will be taken from the verbal phrase at the end of the sentence before the PUNC tagged as Adverb (Adv), or verb (V) having an object mark .

For example the object-verb agreement of the morphology based POS tagged sentence is done as follow.

S2(ኪደነ/N_sm3, ነታ/PRO_sf3, ዝፀረፈቶ/Adj_sf3, ተምሃሪት/N_sf3, ብበትሪ/Adj, ወቒዕዋ/Vp_Ssm3_Osf3, '::'/PUNC_End). From this example the object of the sentence is second word tagged as

ነታ/PRO_sf3 and the verb is the word tagged as object maker ወቒዕዋ/Vp_Ssm3_Osf3.The agreement will be looks like: Agreement_In_sentence = Os+Vs, Of+Vf, O3+V3

4.5.2 Grammar Agreement Relation Checking

This component is responsible for checking whether the input text agreement is grammatically correct or not according to Tigrigna grammar agreement rules after agreement is extracted by MFAE. The system uses the Tigrigna Grammar Rules that can be constructed manually to detect errors in agreement error among noun- modifier, subject-verb, and object-verb, in terms of number, gender, and person; and adverb-verb in time to find the input sentence patterns matches. The matching is done based on linear search of subsequent pattern in the agreement list of the input sentence. If the input agreement pattern exists, the Grammar Agreement Relation Checker considers the input grammar agreement as incorrect agreement relation and it sends the grammar agreement error type and example related to the error type available in the rule to the grammar error filter component. Otherwise, the grammar agreement relation of the input sentence is treated as a correct grammar agreement and sends correct message. The system uses the grammar rule file in the xml file to check errors in a sentence, if a pattern declared in the rule matches the input sentence, then error is shown to the user.

The rules have the properties of rule, pattern, and example. For instance, Figure4.3 shows about noun modifier gender agreement error rule.

```
<rule id="NMA" message="Noun-Modifier gender Agreement Error">
<pattern>Nf+Mm</pattern>
<pattern>Nm+Mf<pattern>
<Example> correct ነዋሕ ላሕሚ, Incorrect ነዊሕ ላሕሚ </Example>
</rule>
```

Figure 4. 3 Sample grammar rule

rule: is the type of agreement error used for automatic testing of its correctness.

pattern: is a list of error tokens that may appear in each agreement rule types.

example: is a string detail of the error type within message displayed to the user along with the rule and error type.

For instance, if the rule based grammar checking finds a match in the rule NMA in the pattern of Nf+Mm or Nm+Mf with the input agreement pattern; then agreement error type “noun modifier gender agreement error”, and the Example “correct ነዋሕ ላሕሚ; incorrect ነዊሕ ላሕሚ “also appear as a sample error.

Our proposed grammar agreement checker algorithm is shown in Algoritm4.8.

```
Read Agreement_In_sentence, rule
Set   error   =   0,   AgreementErrorList[]="",ErrorExample[]="",
ErrorType[]="";
for k = 0; k < Agreement_In_sentence.count; k++
      SET i = 0
      while i <= rule.count
          if Agreement_In_sentence [k] matches rule.pattern [i]
               Add rule.pattern[i]in ErrorList
                Add rule.Example[i]in ErrorExample
               Add rule.type[i] in ErrorType
               error++
           end if
           i++
     end while
End for
if (error>0)
Return ErrorList, Errorexample, ErrorType
else
Return correct
```

Algorithm4. 8Tigrigna Agreement Relation checker

Additionally, if Agreement_In_sentence is given as Sp+Vs, Sm+Vm, S3+V3 in the input sentence (*እቶም/PRO_pm3, ተምሃሮ/N_p,* ናብ/PRE, ትምህርትቤት/N, መጺኡ/Vp_Ssm3,:: /PUN_End) the basic principle of Subject-Verb Agreement is singular subjects need singular verbs; plural subjects need plural verbs; Feminine subject needs Feminine verb, masculine subject needs

masculine verb, and both needs similar person owner etc. Therefore, the subject verb agreement grammar checkers is key point to decide whether the subject and the verb agree or not. In this sentence subject verb agreement agrees in person and gender (Sm+Vm, S3+V3), but disagree in number (Sp+Vs); the subject plural disagree with verbal singular. Thus, a rule is required that matches number, gender and person in the subject and verb. The agreement relation checking detects inputs like this sentence as incorrect text and sends the agreement relation error into the grammar error filtering component, like: Tagged sentence: *ኦቶም/PRO_pm3, ተምሃሮ/N_p,* ናብ/PRE, ትምህርትቤት/N, መጺኡ/Vp_Ssm3, :: /PUN_End

Agreement Error list: Sp+Vs

Error Type: Subject Verb number disagreement

Error Example: *Correct- ተምሃሮ መፂኦም Incorrect- ተምሃሮ መፂኡ*

4.6 Statistical Grammar Checking

The task of rule based grammar checker is limited to detect grammar agreement error types only. However, Tigrigna language has not only SVA, OVA, AVA, and NMA error in grammar but also the WSA error is part of the grammatical error. In order to solve WSA errors, we use statistical based approach to the statistical grammar checker module. Since, the word sequence in a sentence and phrases is not static, it requires large number of rules. It makes difficult to construct rules manually. The idea of using statistical grammar checking as a grammar checker is to analyze an input text and detect any sequence of words that cannot be found in the unique tag sequence and probability language model. The word sequence error in sentences is detected based on Tigrigna word tag sequence probability in the language model. The idea of using this language model as a grammar checker is to analyze an input text and detect any sequence of words that cannot be found in the language model. We conduct N-tag sequence extraction and N-tag sequence checker tasks specifically N-tag is given 3-tag and 2-tag.

The N-tag sequence extraction task is attempted to extract a sequence of tag list by accepting the tagged tokens in a sentence from the grammar agreement checking module. The sequence of tag list is required to identify word sequence grammar by the tag sequence checker. The tag sequence is extracted similar as we use in the Tag sequence gathering module of tag sequence extractor component as 2-tag and 3-tag sequence.

The N-Tag sequence checking is accountable to determines if there is an error in the N-tag sequence list extracted from the input text. This can be done by finding the tag sequence list in the unique tag sequence list, it starts by taking 3-tag and 2-tag sequence list of the tagged tokens; then it finds the probability each 3-tag list from the trigram data in the unique tag sequence list. If each 3-tag list is found the word sequence of the input text is correct. However, if there is any 3-tag list which is not found in the trigram it assigns its probability to 0 and starts checking the input text as 2-tag sequence in the bigram data of the unique tag sequence list. Finally, if all 2-tag sequence lists are found the word sequence agreement of the input text is valid and returns correct message; otherwise the word sequence agreement of the input text is invalid and returns the error 2-tag sequence list to the grammar error filtering module. Algorithm 4.9 is the implementation of the Word sequence checker.

```
Read tagged_Sentence,trigram, bigram
tagseq_List=3tagseqextraction (tagged_Sentence), error_sequence
sentenceProbablity=1
for tagseq in tagseq_List
     if tagseq is in trigram
          sentenceProbablity*= probability of tagseq
     else
          sentenceProbablity*= 0
End for
if sentenceProbablity is 0
tagseq_List=2tagseqextraction (tagged_Sentence)
for every tagseq in tagseq_List
     if tagseq is in Bigram
          sentenceProbablity*= probability of tagseq
      else
           sentenceProbablity*= 0
        Add tagseq in error_sequence
End For
if sentenceProbablity is 0
     return error_sequence
else
     return correct
```

Algorithm4. 9 Tigrigna word sequence error checker

4.7 Grammar Error Filtering

The grammar error filtering module gets the agreement errors of the input text from the rule based grammar checking and statistical grammar checking, and then displays the error type of the sentence, error words, and example related to error type if available. Algorithm4.10 is the implementation of this module.

```
Read AgreementErrorList, Errorseq,
For error in AgreementErrorList
     Display Errortype, Errorwords, and example in error
End for
If Errorseq is correct
     Display error word sequence is none
else
     For errors in errorseq
          Display errorwordseq in errors
     End for
End if
```

Algorithm4. 10 Grammar error filtering

The grammar error filtering now finds the agreement error list having the error type, error word and example parameters and also the error sequence list having the error sequence words. For instance, the Subject verb agreement error in the input (እቶም ተምሃሮ ናብ ትምህርትቤት መጺኡ::) can be filtered as follow: In this sentence, the plural third person subjectተምሃሮ/N_p disagrees with the singular third person verb መጺኡ/V_GER_Ssm3 in number, but there is no word sequence error then after the rule based grammar checking sends the *Agreement ErrorList (Sp+Vs), Error Type: Subject Verb number disagreement, and Error Example:* Correct- ተምሃሮ መፂኦም Incorrect- ተምሃሮ መፂኡ to the grammar error filtering and the grammar agreement filtering displays the error to the user as follow.

Error type: subject verb agreement error in number

Error words: Subject= እቶም, Verb=መጺኡ, Example: Correct- ተምሃሮ መፂኦም Incorrect- ተምሃሮ መፂኡ
Word sequence error: none

Chapter Five: Experiment

5.1 Introduction

In this chapter, we will discuss the corpus preparation of training data and tagger dictionary, the grammar rule preparation, the implementation of the prototype, evaluation of the system, and discussion about the evaluation result. The experimental and evaluation detail of grammatical error checking is also described and then finally the evaluation result of the system's performance is presented.

5.2 Corpus Preparation

In this thesis work, the statistical based grammar checking requires a corpus to train the language model, and the tagging component in the preprocessing module requires a tagger dictionary to assign the input tokens within appropriate tag type. Thus, we use Tigrigna corpus annotated with POS represented in XML structure [17], freely available on the web[1]. The corpus has 4,645 sentences, 72,000 words of 12 categories and 73 tag set. However, within the POS annotated tokens, morpho-syntactic of tokens can represent the more detailed information to detect grammar agreement errors such as gender, person or number agreement. In this study, morpho-syntactic information of each tag set and tagger dictionary is required. Thus, we insert morpho-syntactic information to the POS tag type of the tokens in the corpus manually. The morpho- syntactic information of gender, person, and number is added to Noun, Pronoun, Adjective, Number, and Subject and Object to Verbs. For instance, to the word ሓደረ tagged V_PRF (perfective verb) in the corpus, we insert the subject of the word as V_PRF_Ssm3 (perfective verb, and Subject (singular, masculine, third person).

Finally, tokens in the corpus supports two basic information about a token, its class (noun, verb, adjective, etc.), and detailed morpho-syntactic information (number, gender, person, etc.). Then, we use the morphology-based POS tagged corpus to train the language model. The morphology-based POS tagged corpus is shown in Appendix D.

5.2.1 Dictionary Preparation

The tagger dictionary is prepared from the morphology based POS tagged corpus. The dictionary contains 18,744 unique tokens and their appropriate morphology based POS tags. The tagger

[1] http://eng.jnlp.org/yemane/ntigcorpus

dictionary is prepared in a two columns; the first column is the token list and the second column is the morphology based part-of-speech tag. It is stored in SQL database table as shown in Figure 5.1.

Tag_type	Word_list
N_PRP_pm3	ሃገሮም3
N	ሓገዝ
N_PRP_p	ሓገዝትን
N_pf3	ሓገዝን
V_PRF_Ssm3_C	ሃገርዎ
V_GER_Ssm3_C	ሓገዝዎ
ADJ_s	ሓጋዚ
N_sm3	ሓጋዚኡ
N_PRP_s	ሓጋዝን
N_PRP_s	ሓጋዩን
N_s	ሓጋይ
N_PRP_s	ሓጋይን
V_GER_Ssm2	ሓማሓማኻ
V_IMV_Ssf3	ሓማዝኒ
V_IMV_Ssf3	ሓማዝንን
V_PRF_Ssm2	ሓንሰኻ
N_s1	ሓንሰይ
N_sm3	ሓንሱ
N_s	ሓንስ
N_s	ሓንሶስ
N_PRP_s	ሓንስን
N_PRP_s	ሓንሶስን
N_pm3	ሓንሶሎም
N_PRP_pm3	ሓንሶምን

Figure5. 1 Sample Tigrigna token in the dictionary

5.2.2 Grammar Rule Preparation

The rule based grammar checker uses a grammar rule to analyze grammar agreement error within the Tigrigna input text. We developed a grammar rule based on the Tigrinya grammar book[20].The grammar rules are developed in an XML structure manually categorized into four grammar agreement error types such as MNA, SVA, OVA, and AVA errors. The grammar rule has 114 unique rules. Out of the 114 rules, around 43 rules are MNA, 27 rules are SVA, 27 rules are OVA, and the rest 17 rules are AVA.

For instance, noun modifier rule in Figure 4.3 is prepared based on the following criteria Tigrigna noun phrases agreement rules.

- Singular noun takes singular modifiers
- Plural noun with Plural or common number modifiers
- Common noun with Common modifiers
- Feminine noun with Feminine or Common gender modifiers

- Masculine noun with Masculine or Common gender modifiers

5.3 Implementation

A hybrid Tigrigna Grammar Checker System is implemented using the combination of statistical and rule based approaches where the rule based uses grammatical rules to identify error pattern of input sentences and the statistical uses a single corpus to detect word sequence grammatical errors. In this research work, the prototype used morphology based POS tagged corpus of Tigrigna language, Tigrigna language grammar rule, tagger dictionary, and several development tools.

5.3.1 Developmental Environment and Used Tools

To achieve the research's objective, an essential number of developmental environments are needed. So, the system is developed with the following tools.

Notepad++: is a free tool that is used in this thesis to prepare the annotated Training corpus and agreement grammar rules in an XML structure.

Microsoft Visual Studio: is one of the windows application development platforms, which runs under windows operating system, contains the windows SDK manager, class libraries and SQL database. We use this SDK to create a Graphical User Interface for input of texts from a user and to display errors to the user.

C# programming language: We write a C# code that helps to train the grammar checker algorithm and to acquire experimental results from the database. In order to adopt and train our grammar checker algorithm, we divide the dataset into training and test data.

SharpNlp: is an open source toolkit that contains open source C# modules such as sentence splitter, tokenizer, POS, etc. developed for research development in NLP fields.

SQL database tool: is used to store the tokens within tag type in the tagger dictionary database, and the trained data in the unique tag sequence and probability database that used to determine word sequence agreement error.

The sample input output of the system is shown in Figure 5.2 and detail of the input output of the system is attached in Appendix A.

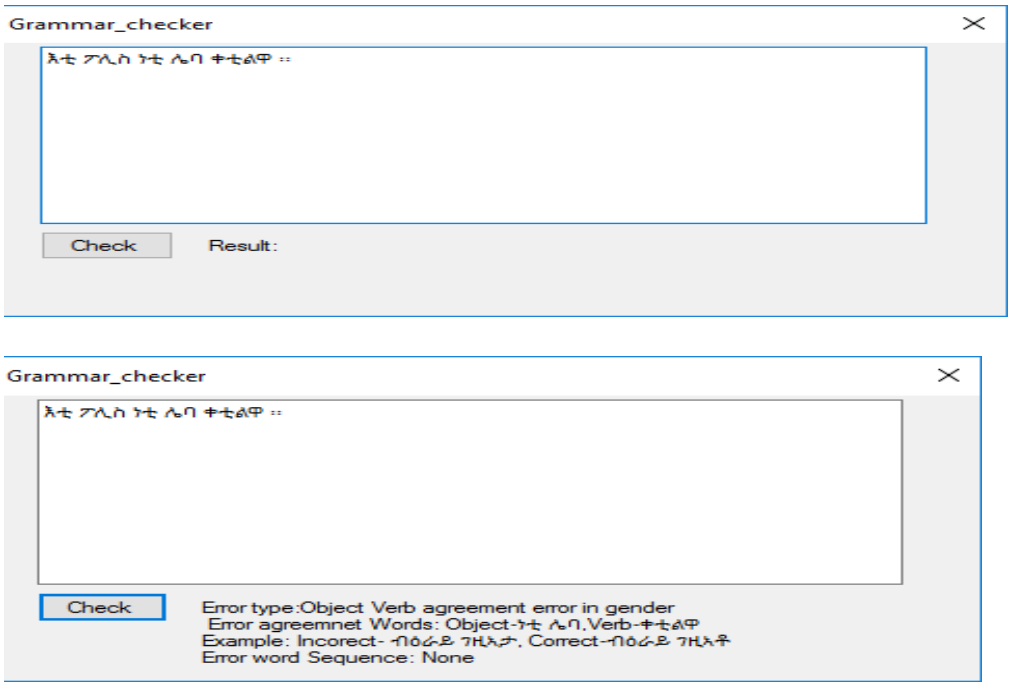


Figure5. 2 Sample input output

5.4 Evaluation

The evaluation of the hybrid Tigrigna Grammar Checker System performance is done towards, to the experiment result of agreement and word sequence grammar errors. Based on the manually counted correct flagged errors, and the correctness, completeness, and accuracy of the system is done with recall, precision and F-score computations. Thus, to evaluate the system we use 300 grammatically correct and incorrect sentences categorized in tothe five grammatical errors as shown in the Table 5.1.

Table5. 1Test data

Grammatical error type	Numbers of flagged correct grammar error	Numbers of wrong flagged grammar errors	Total
SVA	35	15	50
OVA	25	15	40
AVA	20	10	30
NMA	50	30	80
WSA	80	20	100
Total	210	90	300

5.4.1 Evaluation Metrics

In order to evaluate our system performance, we use an evaluation metrics precision, recall, and F-score which are the most commonly used indicators to measure information retrieval and grammar checker performance as we stated in Chapter2.

Thus, the precision, recall, and f-measure in our system can describe as follow. Given that FCE be the number of flagged correct grammar errors and FWE be the number of flagged wrong grammar errors, NFE be the number non flagged grammar errors, then the precision and recall are given by the following equations.

$$\text{Precession} = \frac{\text{CFE}}{\text{CFE}+\text{FWE}} \tag{5.1}$$

$$\text{Recall} = \frac{\text{CFE}}{\text{CFE}+\text{NFE}} \tag{5.2}$$

$$\text{F} - \text{Measure} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5.3}$$

5.4.2 Test Result

The test result of the system is recorded manually on the amount of correctly flagged grammar agreement errors from the 210 Tigrigna correct grammar error sentences and 90 wrong grammar error sentences. To do so, we perform system testing in four different days by labeling as Exp1, Exp2, Exp3, and Exp4 of results in day1, day2, day3, and day4 respectively. In each experiment seventy five (75) sentences were distributed and tested daily and accordingly, the results of correctly flagged grammar agreement errors are shown in the Table 5.2 and Figure 5.3.

Table5. 2 Correctly flagged result in each experiment

Agreement Error type	Experiments result				
	Exp1	Exp2	Exp3	Exp4	Average
Subject Verb agreement	78%	72%	80%	82%	78%
Object Verb agreement	83%	81%	79%	76%	79.8%
Noun Modifier agreement	89%	82%	89%	86%	86.5%
Adverb Verb agreement	72%	77%	76%	74%	74.8%
Word Sequence agreement	82%	76%	78%	80%	79%
Average Experiment result	80.8%	77.6%	80.4%	79.6%	79.6%

The Average experiment result of correctly flagged grammar error type is shown Figure 5.3.

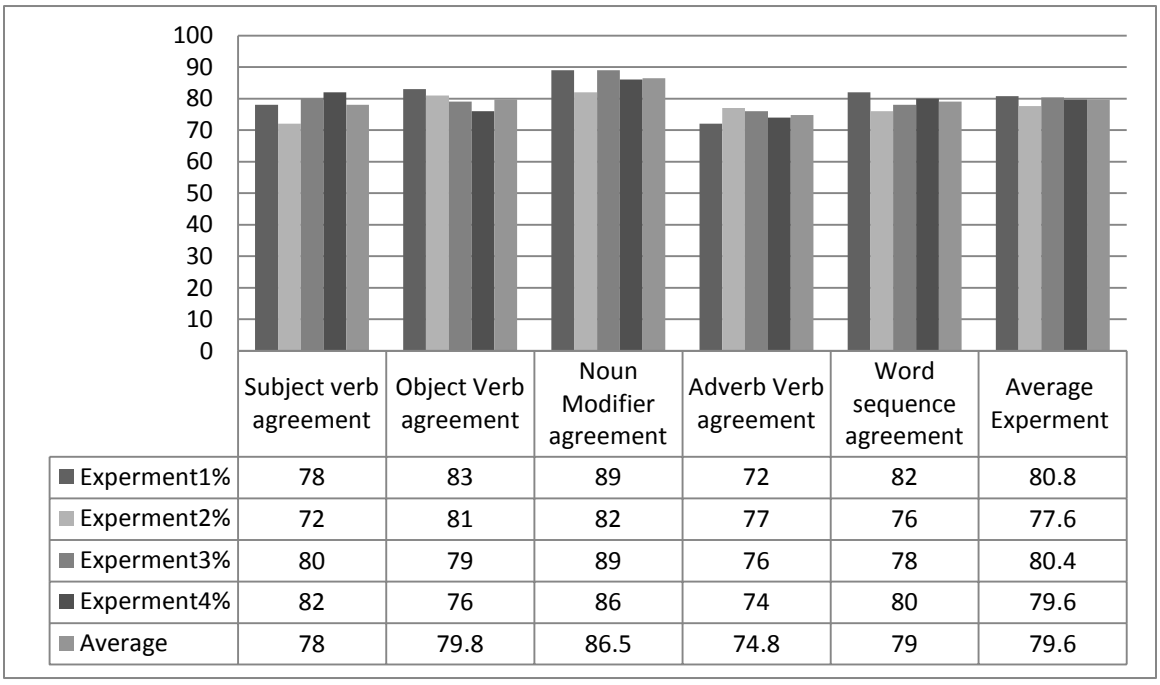


Figure5. 3 Average experiment result

To measure the performance of our system, we use the precision, recall and F-score metrics based on the average experiment result of the grammar error types. Table5.3 describes the performance of Tigrigna Grammar checker system.

Table5. 3Average Performance of Hybrid Tigrigna Grammar Checker System

Grammar Error Type	Number of input grammar error	Correct Flagged Grammar Errors	Wrong Flagged Grammar errors	Not flagged Grammar errors	Precision	Recall	F–measure
Subject-verb	50	39	7	4	84.8	90.7	87.7
Object-Verb	40	32	3	5	92	86	88.9
Noun-Modifier agreement	80	70	3	7	95.6	90.6	93
Adverb Verb agreement	30	23	5	2	82.4	79.8	81
Word-sequence agreement	100	79	11	10	87.8	88.7	88.2
Total	300	243	29	28	84.7	89	86.8
Average					87.9	87.5	87.6

The average evaluation result of the Hybrid Tigrigna Grammar Checker System is shown in Figure 5.4 based on recall, precision and f-measure evaluation metrics.

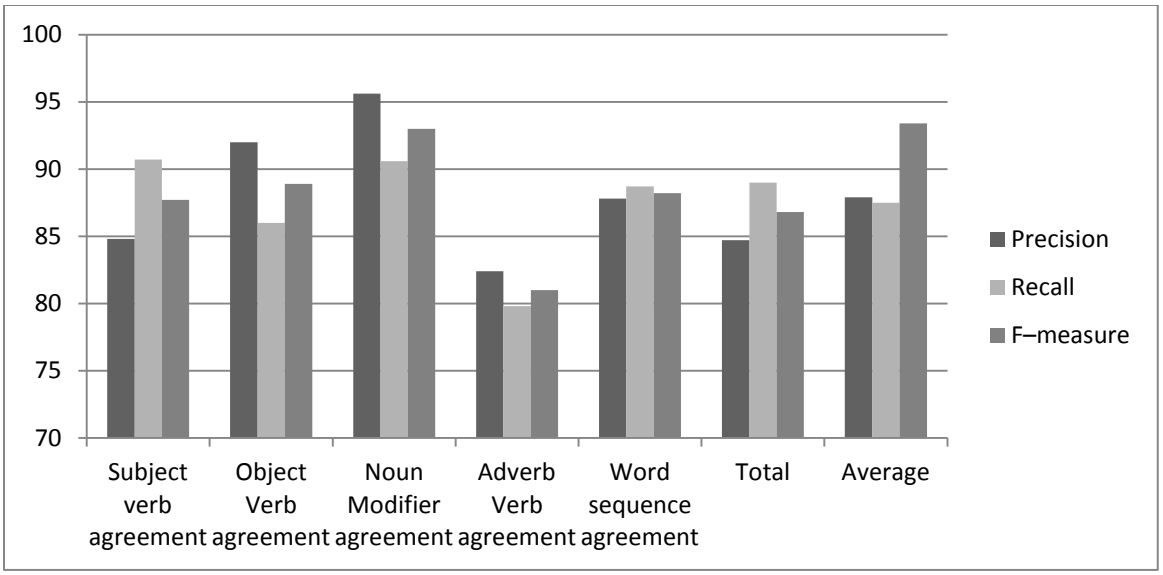


Figure5. 4 Average performance of Hybrid Tigrigna Grammar Checker System

5.5 Discussion

The experiment is conducted manually to evaluate the performance of the system recorded against the test data of 300 sentences in four different experiments input output result as it is shown in the Table5.2.The average performance result shows significant outcome of 87.9% precision, 87.5% recall, and 87.6% f-measure to determine the test data's grammatical agreement result as it is shown in Figure 5.3. However, the results recorded from the experiments are less due to two main reasons. The first reason is the tagger; the tagger in this system uses a manual dictionary words which do not occur in the dictionary are tagged as "Unknown". Since the system uses the tag pattern information then texts having "Unknown" tags are determined as incorrect word sequence agreement errors. Second, incorrect texts having dependent sentences are examined as correct, this problem occurred because the system splits texts into sentences using the sentence end punctuations. Then, it checks for an error in the sentences independently, for instance, the input text አበበ ን ትምህርቲ ምስ ከደ፡፡ ስራሕ እውን ጀማሮም። (After Abebe goes to education. They also started work) in this text the verb ጀማሮ ም/ V_GER_Spm3 in the second sentences the actions of the subject አ በ በ /N_sm in the first sentence disagrees in number which is the subject is singular and verb is plural (Ss+Vp). But the system examines error within independent sentences; it categorizes such error texts as grammatically correct.

Chapter Six: Conclusion and Future Work

6.1 Conclusion

Grammar checker system is one of applications in NLP. The system is developed to minimize time and cost to examining grammatically incorrect written texts compared to using manually checking. To meet the objectives of a grammar checker system, many studies have been attempted to several languages depend on the linguistic structure and development approaches. However, as the best knowledge of the researcher, there is no study conducted to develop Tigrigna grammatical checker system.

The aim of this thesis is to attempt the development of Hybrid Tigrigna Grammar Checker System that applies to detect Tigrigna text grammatical agreement errors. The rule based grammar checker is responsible to detect four types of grammar agreement errors in a sentence such as SVA, OVA, AVA, and MNA using manually developed 114 rule sets. Whereas the statistical grammar checker is done to determine word sequence grammar agreement errors using the tag sequence probabilities which are 2-tag sequence and 3-tag sequence statistical result of 4645 morphology based POS annotated corpus.

We have tested the system performance using manually prepared incorrect and correct flagged grammar errors of 300 sentences. The system successfully detects most of the sentence with incorrect and correct grammar agreement errors. But, some dependent sentences having grammatical errors are determined as correct, in addition to the dependent sentences, correct sentence having words not found in the tagger dictionary also are determined as incorrect sentences. However, the overall performance of our system shows significant result to detect independent sentences by performing 87.9% precision, 87.5% recall, and 87.6% f-measure.

6.2 Contribution of the Work

The most relevant contributions of this thesis work are listed as follows:

- The study has adopted other hybrid based grammar checker systems to agreement error and word sequence error in a sentence of Tigrigna language. This can contribute for other researchers related to Tigrinya as a reference.
- The study has developed rules to detect grammar agreement error in a Tigrigna language sentence. This can contribute as an input for other researchers who want to deal with grammar checker exhaustively in the language.

- The study has developed a dictionary of 18,744 unique words annotated with morphology and POS. This can help for lexicographers as an insight and develop a comprehensive electronic dictionary.

6.3 Future Work

We have tried to cover as many grammar error types as we could to solve an error within a sentence in any input of Tigrigna text. However, there are many types of grammar error that probably occur in a sentence are not covered. Hence we can recommend working on the following research works:

- We have used only 18,744 unique tokens of POS and Morphology information. We suggest that better result can be accomplished by adding automatic spelling checker, POST, and morphological analyzer could be maximizing the performance of the grammar checker in which the word information is extracted in a better way.
- This research is done to solve grammar error texts within independent sentence that has sentences end punctuation markers. Thus, to develop a grammar checker to solve grammar error in a paragraph level and sentence relation error with in a sentence is an open research area.
- Developing the rules to decompose the compound complex and compound sentence into simple and complex sentence using boundary splitter and sentence checker.

References

[1] R. Bears and N. Cloud, “Language Group Specific Informational Reports”, in *M.Ed. In TESL Program*, Rhode Island College, 2011

[2] J. Sie and Y. Le, “Automatic Correction of Grammatical Errors in Non-native English Text”, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 2009.

[3] J Bigert, Automatic and Unsupervised Methods in Natural Language Processing, KTH Numerisk analys och datalogi, Stockholm, Sweden, 2005.

[4] N.S. Bhirud, R.P. Bhavsar, and B.V. Pawar ,“Grammar Checkers for Natural Languages”, in *International Journal on Natural Language Computing (IJNLC) Vol. 6, No.4, August 2017*

[5] M. Konchady, "Detecting Grammatical Errors in Text using a Ngram-based Ruleset," in *RMV Extension* Bangalore, India.

[6] M. J. Alam, N. UzZaman and M. Khan, "N-gram based Statistical Grammar Checker for Bangla and English," in *Center for Research on Bangla Language Processing* Dhaka, Bangladesh.

[7] N.Y. Lin, K. M. Soe, and N. L. Thein, “Chunk-based Grammar Checker for Detection Translated English Sentences”, in *International Journal of Computer Applications (0975 – 8887) Volume 28, No.1, August 2011.*

[8] M. Mozgovoy,” Dependency-Based Rules for Grammar Checking with LanguageTool”, in *Proceedings of the Federated Conference on Computer Science and Information Systems pp. 209–212.*

[9] Kidane Mebrahtu, "Morphological Analyzer for Tigrigna Nouns and Adjectives with Finite State Transducer Techniques," Department of Computer Science, University of Gondar, July 2015.

[10] Haylay Beyene, “Design and Development of Tigrigna search engine”, Department of computer science, Addis Ababa University, 2013

[11] J. Xing, L. Wang, D. F. Wong, L. S. Chao, and X.Zeng, "A Hybrid System for English Grammatical Error Correction," *Computational Natural Language Learning: Shared Task,* vol. 7, pp. 34-42, August 2013.

[12] R. Domeij, O. Knutsson, J. Carlberger, and V. Kann, "Granska an efficient hybrid system for Swedish grammar checking," in *Proceedings of NODALIDA,* Stockholm, Sweden, 1999.

[13] J. Carlberger,R. Domeij, V.Kann and O.Knutsson ,"The Development and Performance of a Grammar Checker for Swedish", *Cambridge University Press*, Stockholm, Sweden, 2004.

[14] S. Cheng, C. Yu, and H. Chen, "Chinese Word Ordering Errors Detection and Correction for Non-Native Chinese Language Learners", in *proceeding of COLING the 25th International Conference on Computational Linguistics: Technical Papers*, Dublin, Ireland, 2014, pp279–289.

[15] Aynadis Temesgen and Yaregal Assabie, "Design and Development of Amharic Grammar Checker", Department of Computer Science, Addis Ababa University, March 2013.

[16] Debela Tesfaye, "A rule-based Afan Oromo Grammar Checker," *(IJACSA) International Journal of Advanced Computer Science and Applications*, vol. 2, pp. 126-130, 2011.

[17] Yemane Keleta, K. Yamamoto, and A. Marasinghe, "Nagaoka Tigrinya Corpus: Design and Development of Part-of-speech Tagged Corpus", *The Association for Natural Language Processing*, 2016.

[18] T Holan, V Kubofi, and M Plfitek, "A Prototype of a Grammar Checker for Czech", Language Technologies for Slavic Languages, No.85/1995, Charles University, Czech Republic.

[19] Teklay Gebregzabiher, "Part of Speech Tagger for Tigrigna Language," Department of Information Science, Addis Ababa University, November 2011.

[20] Dan'el Taklu, *Zemenawi sewasew quanqa Tigrigna*, Mega Press, Addis Ababa, 2012.

[21] Yonas Fisseha, "Development of Stemming Algorithm for Tigrigna text", Department of Information Science, Addis Ababa University, 2011.

[22] What is grammar, *http://www.linguisticsociety.org/sites/default/files/Grammar.pdf*, (accessed in Aug 2015)

[23] K. M. Wilcox, "Defining Grammar a Critical Primer", Unpublished Master's Thesis, Department of Master of Arts in English, Monatana State University, April 2004.

[24] V. Henrich and T. Reuter, "LISGrammarChecker", Unpublished Master's Thesis, Department of Computer Science, Hochschule Darmstadt and Reykjavík University, Feb 2009.

[25] B. K. Bal and P. Shrestha, "Architectural and System Design of the Nepal Grammar Checker", *Madan Puraskar Pustakalaya*, PatanDhoka, Nepal.

[26] D. Jurafsky and J. H. Martin, "N-GRAMS," *Speech and Language Processing*, January 2007, pp. 1-13.

[27] M. Mittal, D. Kumar, and S. K. Sharma, "Grammar Checker for Asian Languages", *International Journal of Computer Applications and Information Technology*, vol. 9, 2016, pp. 163-167.

[28] M. S. Gill and G. S. Lehal, "A Grammar Checking System for Punjabi," in *Posters and Demonstrations,* Manchester, August 2008.

[29] Daniel Naber, "A Rule-Based Style and Grammar Checker", Bielefeld University, 2003

[30] P. S. Kernick and D. W. Powers, "A Statistical Grammar Checker," *Flinders University of SA,* August 1996.

[31] I. Markov, "Information Retrieval Offline Evaluation", University of Amsterdam, Netherlands.

[32] L Derczynski, "Complementarity, F-score, and NLP Evaluation", University of Sheffield, UK.

[33] J. Makhoul, F. Kubala, R. Schwartz, and R. Weischedel "Performance Measures for Information Extraction", *BBN Technologies, GTE Corp., Cambridge, MA 02138*

[34] S. M. Cheng, C. H. Yu, and H. H. Chen, "Chinese Word Ordering Errors Detection and Correction for Non-Native Chinese Language Learners", Department of Computer Science and Information Engineering National Taiwan University, Taipei, Taiwan, 2014.

[35] Ge'ez script, http://www.omniglot.com/writing/ethiopic.htm (accessed Nov 7, 2015)

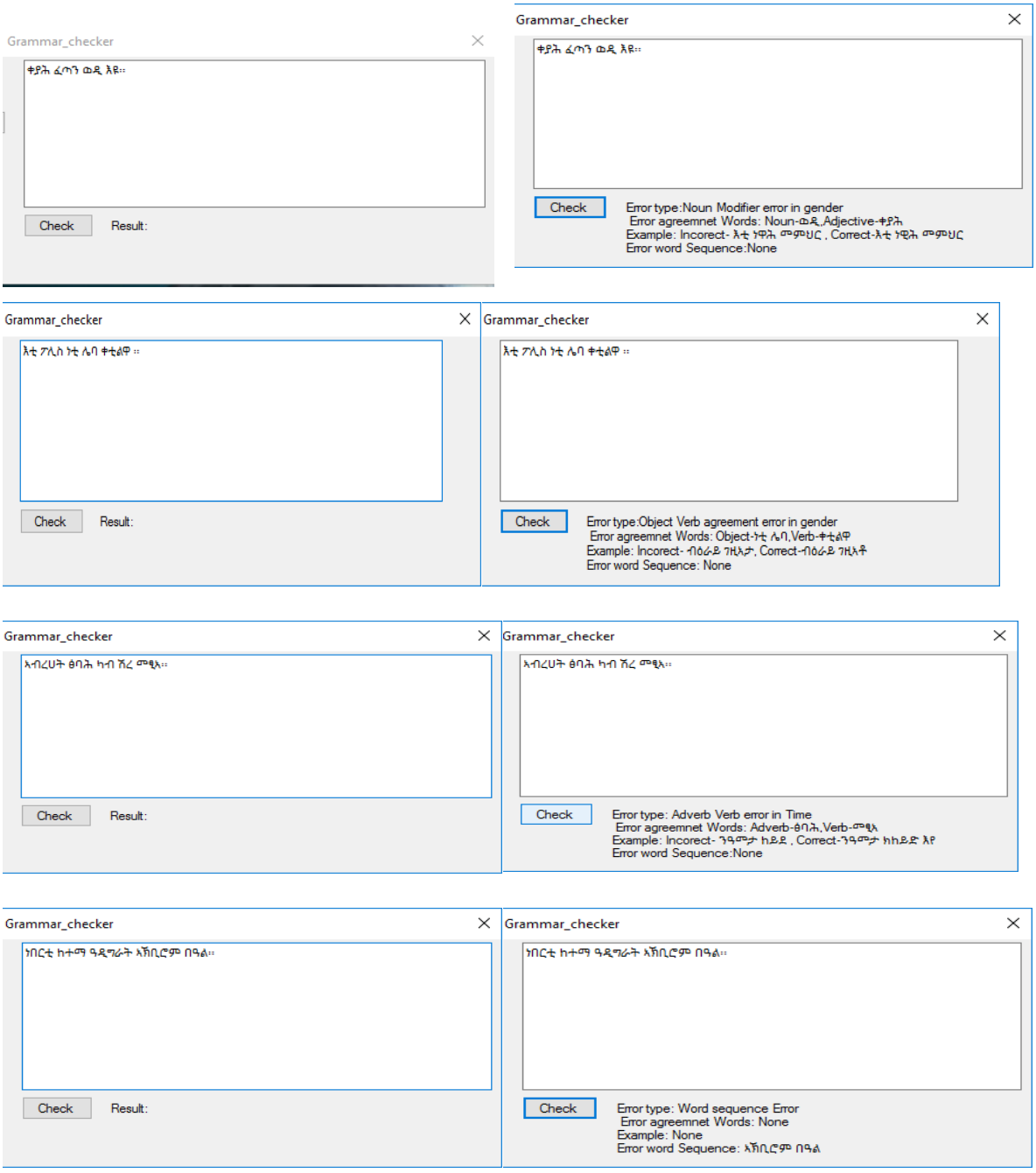
[36] P J. McAlexander , "*Checking the Grammar Checker: Integrating Grammar Instruction With Writing*", journal of Basic Writing,Vol.19, No.2, 2000.

[37] P S Kernick and D M Powers, "A Statistical Grammar Checker", Department of Computer Science, Flinders University of SA, 1995.

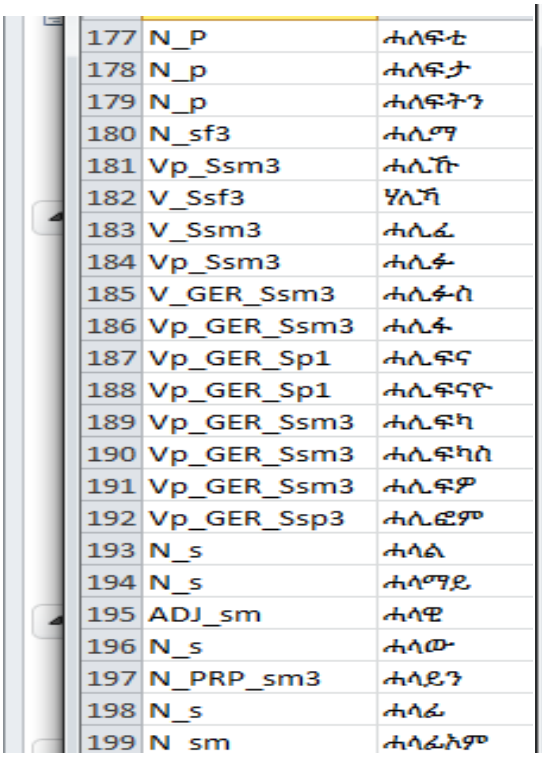
[38] M. Gasser, "Semitic morphological analysis and generation using finite state transducers with feature structures," in *Conference of the European Chapter of the Association for Computational Linguistics,* Vol. 12.

Appendix A: Sample Input Output in System Testing

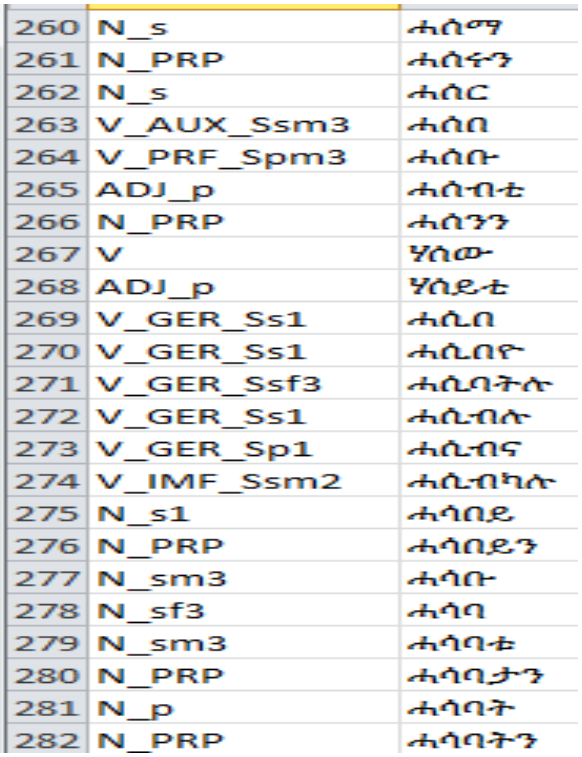
Input	Output


Grammar_checker
ቀያሕ ፈጣን ውዲ እዩ።
Check
Result:
Grammar_checker
ቀያሕ ፈጣን ውዲ እዩ።
Check
Error type:Noun Modifier error in gender
Error agreemnet Words: Noun-ውዲ,Adjective-ቀያሕ
Example: Incorect- እቲ ነዋሕ መምህር , Correct-እቲ ነዊሕ መምህር
Error word Sequence:None
Grammar_checker
እቲ ፖሊስ ነቲ ሌባ ቀቲልዋ ።
Check
Result:
Grammar_checker
እቲ ፖሊስ ነቲ ሌባ ቀቲልዋ ።
Check
Error type:Object Verb agreement error in gender
Error agreemnet Words: Object-ነቲ ሌባ,Verb-ቀቲልዋ
Example: Incorect- ብዕራይ ገዚአታ, Correct-ብዕራይ ገዚአቶ
Error word Sequence: None
Grammar_checker
አብረሀት ፅባሕ ካብ ሽረ መፂኡ።
Check
Result:
Grammar_checker
አብረሀት ፅባሕ ካብ ሽረ መፂኡ።
Check
Error type: Adverb Verb error in Time
Error agreemnet Words: Adverb-ፅባሕ,Verb-መፂኡ
Example: Incorect- ንዓመታ ከይደ , Correct-ንዓመታ ክከይድ እየ
Error word Sequence:None
Grammar_checker
ነበርቲ ከተማ ዓዲግራት ኣኽቢሮም በዓል።
Check
Result:
Grammar_checker
ነበርቲ ከተማ ዓዲግራት ኣኽቢሮም በዓል።
Check
Error type: Word sequence Error
Error agreemnet Words: None
Example: None
Error word Sequence: ኣኽቢሮም በዓል

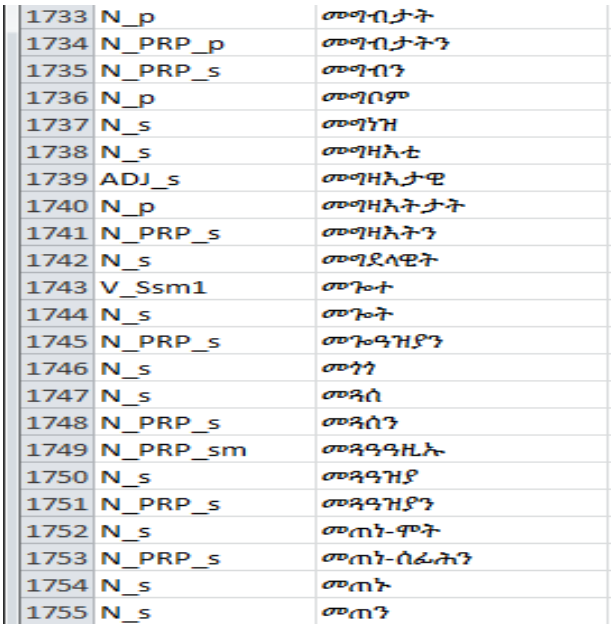
Appendix B: Sample Morphology based tagged words in the Tagger Dictionary



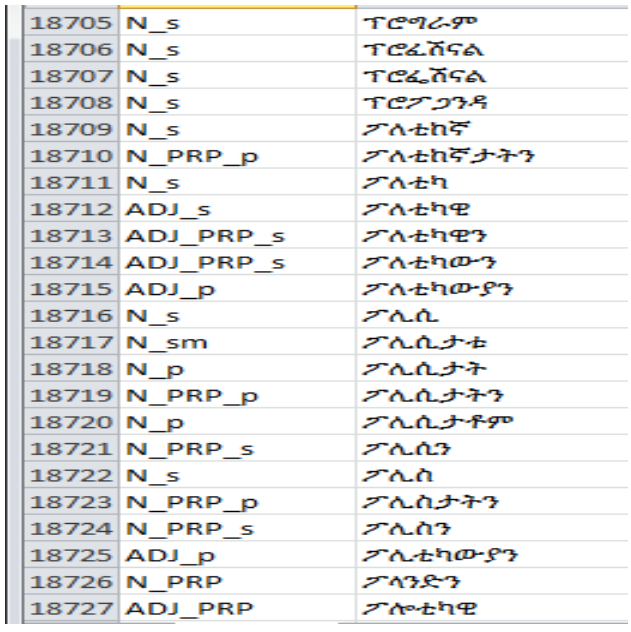
177	N_P	ሐለፍቲ
178	N_p	ሐለፍታ
179	N_p	ሐለፍትን
180	N_sf3	ሐሊማ
181	Vp_Ssm3	ሐሊኹ
182	V_Ssf3	ሃሊካ
183	V_Ssm3	ሐሊፈ
184	Vp_Ssm3	ሐሊቆ
185	V_GER_Ssm3	ሐሊቆስ
186	Vp_GER_Ssm3	ሐሊቆ
187	Vp_GER_Sp1	ሐሊፍና
188	Vp_GER_Sp1	ሐሊፍናዮ
189	Vp_GER_Ssm3	ሐሊፍካ
190	Vp_GER_Ssm3	ሐሊፍካስ
191	Vp_GER_Ssm3	ሐሊፍዎ
192	Vp_GER_Ssp3	ሐሊፎም
193	N_s	ሐላል
194	N_s	ሐላማይ
195	ADJ_sm	ሐላዊ
196	N_s	ሐላው
197	N_PRP_sm3	ሐላይን
198	N_s	ሐላፊ
199	N_sm	ሐላፊአም



260	N_s	ሐሰማ
261	N_PRP	ሐሰሩን
262	N_s	ሐሰር
263	V_AUX_Ssm3	ሐሰበ
264	V_PRF_Spm3	ሐሰቡ
265	ADJ_p	ሐሰብቲ
266	N_PRP	ሐሰንን
267	V	ሃሰው
268	ADJ_p	ሃሰይቲ
269	V_GER_Ss1	ሐሲበ
270	V_GER_Ss1	ሐሲበዮ
271	V_GER_Ssf3	ሐሲባትሉ
272	V_GER_Ss1	ሐሲብሉ
273	V_GER_Sp1	ሐሲብና
274	V_IMF_Ssm2	ሐሲብካሉ
275	N_s1	ሐሳበይ
276	N_PRP	ሐሳበይን
277	N_sm3	ሐሳቡ
278	N_sf3	ሐሳባ
279	N_sm3	ሐሳባቱ
280	N_PRP	ሐሳባታን
281	N_p	ሐሳባት
282	N_PRP	ሐሳባትን



1733	N_p	መግብታት
1734	N_PRP_p	መግብታትን
1735	N_PRP_s	መግብን
1736	N_p	መግቦም
1737	N_s	መግኝዝ
1738	N_s	መግዛኢቲ
1739	ADJ_s	መግዛኢታዊ
1740	N_p	መግዛእትታት
1741	N_PRP_s	መግዛእትን
1742	N_s	መግደላዊት
1743	V_Ssm1	መጐተ
1744	N_s	መጐት
1745	N_PRP_s	መጐዓዝያን
1746	N_s	መጎጎ
1747	N_s	መጓሰ
1748	N_PRP_s	መጓሰን
1749	N_PRP_sm	መጓዓዓዚኡ
1750	N_s	መጓዓዝያ
1751	N_PRP_s	መጓዓዝያን
1752	N_s	መጠነ-ሞት
1753	N_PRP_s	መጠነ-ሰፊሕን
1754	N_s	መጠኑ
1755	N_s	መጠን



18705	N_s	ፕሮግራም
18706	N_s	ፕሮፌሽናል
18707	N_s	ፕሮፌሽናል
18708	N_s	ፕሮፓጋንዳ
18709	N_s	ፖለቲከኛ
18710	N_PRP_p	ፖለቲከኛታትን
18711	N_s	ፖለቲካ
18712	ADJ_s	ፖለቲካዊ
18713	ADJ_PRP_s	ፖለቲካዊን
18714	ADJ_PRP_s	ፖለቲካውን
18715	ADJ_p	ፖለቲካውያን
18716	N_s	ፖሊሲ
18717	N_sm	ፖሊሲታቱ
18718	N_p	ፖሊሲታት
18719	N_PRP_p	ፖሊሲታትን
18720	N_p	ፖሊሲታቶም
18721	N_PRP_s	ፖሊሲን
18722	N_s	ፖሊስ
18723	N_PRP_p	ፖሊስታትን
18724	N_PRP_s	ፖሊስን
18725	ADJ_p	ፖሊቲካውያን
18726	N_PRP	ፖላንድን
18727	ADJ_PRP	ፖሎቲካዊ

Appendix C: Grammar Rule Structure

```
<rules>
<rule id="SVA" message="Subject-Verb Agreement Error">
<rule id="OVA" message="Object-Verb Agreement Error">
<pattern id="Gender">
<pattern id="Number">
<pattern id="Person">
</rule>
<rule id="NMA" message="Noun-Modifier Agreement Error">
<rule id="AVA" message="Adverb-Verb Agreement Error">
</rules>
```

Appendix D: Sample Morphology based POS Tagged Corpus

```
<s n="1"> <w type="ADJ_s">አእምሮኣዊ</w> <w type="N_s">ጥዕና</w> <c type="PUN_end">።</c> </s>
<s n="2"> <w type="PRE">ቅድሚ</w> <w type="ADJ_p">ብዙሕ</w> <w type="N_p">ዓመታት</w> <c type="PUN_dcot">"</c>
<w type="ADJ_s">አእምሮኣዊ</w> <w type="N_s">ስንክልና</w> <w type="N_PRP_s">ብጋኔን</w> <w type="CON">ወይ</w>
<w type="ADJ_s">እከይ</w> <w type="N_p">መናፍስቲ</w> <w type="V_AUX_Ssm3">ኢዩ</w> <w type="V_REL_Ssm3">ዝመጽእ</w>
<c type="PUN_dcot">"</c> <w type="V_REL_Ssm3">ዝብል</w> <w type="ADJ_s">ግጉይ</w> <w type="N_s">ኣመለኻኽታ</w>
<w type="V_GER_Ssm3">ነይሩ</w> <c type="PUN_end">።</c> </s>
<s n="3"> <w type="CON">ከም</w> <w type="N_sm3">ውጺኢቱ</w> <w type="CON">ድማ</w> <w type="ADJ_s">አእምሮኣዊ</w>
<w type="N_s">ስንክልና</w> <w type="ADJ_p">ዘጋጠሞም</w> <w type="N_p">ኣባላት</w> <w type="PRE">ናይ</w>
<w type="NUM">ሓደ</w> <w type="N_s">ሕብረተ-ሰብ</w> <w type="ADJ_PRP_s">ብኣሰቃቕኝ</w> <w type="ADJ_PRP_s">ኢሰብኣውን</w>
<w type="N_s">ኣገባብ</w> <w type="V_IMF_Spm3">ይተሓዙ</w> <w type="V_AUX_Spm3">ነይሮም</w> <c type="PUN_end">።</c> </s>
<s n="4"> <w type="ADJ_s">ጸቢብ</w> <w type="PRE">ኣብ</w><w type="ADJ_p">ዘይኮኑ</w> <w type="N_p">ከባቢታት</w>
<w type="CON">እውን</w> <w type="ADJ_sm">እዚ</w> <w type="N_s">ኣመለኻኽታ</w> <w type="ADJ_sm">እዚ</w> <w type="V_IMF_Ss">ይስርሓሉ</w>
<w type="N_V_s">ምህላዉ</w> <w type="V_IMF_Ssm">ይማልጽ</w> <w type="V_AUX_Ssm3">እዩ</w> <c type="PUN_end">።</c> </s>
<s n="5"> <w type="PRO">ገለ</w> <w type="PRO_p">ካብቶም</w> <w type="PRO_s">ካብቲ</w> <w type="ADJ_s">ሕሱም</w>
<w type="N_s">ኣገባብ</w> <w type="N_s">ኣተሓሕዛ</w> <w type="ADJ_p">ዝመሰቑ</w> <w type="N_p">ማጻያት</w>
<w type="CON">እውን</w> <w type="PRE">ናብ</w> <w type="N_p">በረኻታት</w> <w type="V_PRF_Spm3">እናሃደሙ</w>
<w type="PRO_pm3">በይናቶም</w> <w type="V_IMF_Spm3">ይነብሩ</w> <w type="CON">ወይ</w> <w type="CON">እውን</w>
<w type="N_s">ድራር</w> <w type="N_s">ኣራዊት</w> <w type="V_IMF_Spm3">ይኾኑ</w> <w type="V_AUX_Spm3">እዮም</w>
<c type="PUN_end">።</c> </s>
```

Appendix E: Sample Test Corpus

Sentence	Error Type	Error Words	System result
ነቲ ጉዕዞ ፍቕሪ ተታሓሒዝዎ ዝስዕብ።	Word sequence	ዝስዕብ ።	✓
ኣብ ዞባ ማእከል ብሸለልትነት ዝስዕቡ ሓደጋታት ብንቕሓት ንምግትኡ ዝካየድ ጻዕሪ ክሕይል መዘኻኸሪ ቀሪቡ፡፡	Correct		✓
ኣባል ኣወሃሃዲ ሽማግለ በዓላት ዞባ ደቡብ ኣቶ ኣብርሃም ዮውሃንስ ፅባሕ ካብ ጀርመን ከይዱ፡፡	Adverb-Verb	ፅባሕ ከይዱ	✓
ኣቶ ኣብርሃም ፈስቲቫል ዝካየደሉ ቦታ ኣብሪሁ።	Correct		✓
ጸሊም ሕብሪ መግለጺ ምእዙዝነትን ትእዛዝን።	Word sequence	ትእዛዝን ።	✓
ከበደ ነቶም ኣብዑር ትማሊ ካብ ዓዲ ኣምፂእዋ፡፡	Object-Verb	ኣብዑር ኣምፂእዋ	✓
ዝበዝሐ እዋን ከም መግለጺ ፍቕሪ ዝግለጽ። ንባህሪ ዝውክል ዝግለጽ ቀጠልያ ሕብሪ።	correct		X
ብድኽነት ዝሳቐዩ ዝነበሩ ቁልዑ 2 ሚልዮን እዩ ።	Subject-Verb	ቁልዑ እዩ	✓
ኣብ ብሪጣንያ ዝተኻየደ መጽናዕቲ ኣብ ጋዜጣ ዘጋርድያን ተሓቲሙ ፡፡	Correct		✓
እዚ ክልተ ጥበባት እዚ ዝተመርጸሉ ቀንዲ ምኽንያት እቲ ካናለ ጥውይዋይን ቀላጥ ቦታን ስለ ።	Word sequence	ስለ ።	✓
እቶም ተማሃሮ ማይን ሓመድን ንምዕቃብ ቃል ኣትዩ፡፡	Subject-Verb	ተማሃሮ ኣትዩ	✓
ተማሃሮ ካልኣይ ደረጃ ኣብያተ-ትምህርቲ ንማእቶት ኣዝዮም እዮም ዝጽበየኒ።	Object-Verb	ንማእቶት ዝጽበየኒ	X
ኣእምሮኣዊ ስንክልና ክንፈልጦን ብኡ ኣቢልና ድማ ቅኑዕ ኣገባብ ኣተኣላልያ ናይ ኣእምሮ ሕሙማት ኽንከተልን አለና፡፡	Correct		✓
ኣብ ዘይኮነ ከባቢታት ይግልጽ እዩ።	Word sequence	ከባቢታት ይግልጽ	✓
ናይ አፍሪካ ሕብረት መረሕቲ መዮበኛ ስብሰባ ኣበይ ተካይደ?	Correct		✓
እቲ ወትሩ እትገብሮ ጉያ ብግዜ ንኽትርእዮ ዕድል ዝህበካ ኣይኮነን ።	Correct		✓
ፕረዚደንት እቲ ማሕበር ወይዘሮ ኣዜብ ተወልደ። ኣብ ዝሓለፈ ዓመታት ዲጂታላዊ ቤት-ንባብ ንምትእትታው ዝወጸ መደብን ዘጋጠሞ ፅንቅፋትን ንኣባላት ዝተዋህቡ ስልጠናታት ዝተወደቡ ዓውደ-መጽናዕትታትን ካልእን ዝምልከት ጸብጻብን ናይ መጻኢ መደባትን ኣቕሪባ ።	Correct		X
ምሁራን ብዙሕ ነገር ክምህሩ ይኽእሉ እዩ።	Subject-Verb	ምሁራን እዩ	✓
ህዝቢ ኢዩ ስልጣን ዝህብ።	correct		✓

ኮሚሽን ስፖርትወርሓዊ ገምጋም ንጥፈታት ስፖርት ዞባታት ኣካይዱ ።	Correct		✓
እቲ ትግርኛ ዝዛረብ ፖሊስ ኢትዮጵያዊ እዩ ።	Correct		✓
ደቀንስትዮ ዓው ኢልካ። ንምሕሳብ ብዓይኒ ምሕዝነትን ናጻ ኰንካ ምቕራብን እየን ዝርእየኣ ።	Correct		X
ደቂተባዕትዮ ተዋጋእቲ ተኸላኸልትን ፈታሕቲ ጸገማትን ።	Word sequence	ጸገማትን ።	✓
ብርሃነ ፅባሕ ናብ መቐለ መሚኡ ።።	Adverb –Verb	ፅባሕ መሚኡ	✓
እቶም መንእሰያት ዝገፍሐ መሬት ናብ ንቡር ክመልሱን ናብ ሓምላይ ክቕይሩን ትጽቢት ተነቢርሎም ኣሎ ።	Correct		✓
በዓል መስቀል ትማሊ 27 መስከረም ኣብ መላእ ሃገር ብድምቀት ተኸቢሩ ።	Correct		✓
ነዋሕ ወዲ እዩ።	Noun-Modifier	ነዋሕ ወዲ	✓
ከተማ ሐረር ኪል ሜትር ትርሕቅ?	Word sequence	ሐረር ኪል	✓
ዓቕሊ ዋጋ በቕሊ!	Correct		✓
እቶም ቅብሊታት ንጀመርቲ መስተኣናገድቲ ኣድላዩ እዩ።።	Subject-Verb	ቅብሊታት እዩ	✓
ባይቶ ስፖርት ኣፍሪቃ ሽልማት ሂቡ ።	Correct		✓
ኣብ ትምህርቲ ልዑል ተገዳስነት ኣለኒ ትምህርቲ።	Word sequence	ኣለኒ ትምህርቲ	✓
ካብ ፍርቂ ለይቲ ንደሓር ደቂሰ እንከለኹ። ሃተፍተፍ ክብል እሓድር ።	Correct		✓
ኣብ ዕብየት ቋንቋ ንእሽቶ ዝበሃል ኣበርክቶ የለን ።	Correct		✓
ጸባ ጡብ ኣደ ዛጊት እቲ ዝበለጸን እቲ ዝበለጸን መተካእታ ዘይተረኽቦን መግቢ ህጻን እዩ ።	Word sequence	ዝበለጸን እቲ	✓
ናይ ኢትዮጵያ ኣትላቲክስ ፌዳሬሽን ዓመታዊ።	Word sequence	ዓመታዊ።	✓

Appendix F: Sample Bigram and Trigram data

Bigram	
Two_tag_sequence	**Probability**
ADJ,N	0.01486989
N,PUN_End	0.01115242
PRE,N	0.007434944
N,N_p	0.01115242
N_p,PUN_dcot	0.007434944
PUN_dcot,ADJ	0.01115242
ADJ,N_s	0.04089219
N_s,N_PRP_s	0.01115242
N_PRP_s,N_s	0.01115242
N_s,AD_s	0.01115242
AD_s,N_p	0.01115242
N_p,V_AUX_Ssm3	0.01115242
V_AUX_Ssm3,ADJ	0.01115242
ADJ,PUN_dcot	0.01115242
PUN_dcot,V_REL	0.01115242
V_REL,ADJ	0.01115242
ADJ,N_s	0.04089219
N_s,V_AUX_Ssm3	0.007434944
V_AUX_Ssm3,V_IMV	0.007434944
V_IMV,N_sm3	0.007434944
N_sm3,N_PRP_s	0.007434944
N_PRP_s,ADJ	0.01115242
ADJ,N_s	0.04089219
N_s,ADJ_p3	0.007434944
ADJ_p3,N	0.007434944
N,N_PRP	0.007434944

Trigram	
Three_Tag_sequence	**Probability**
ADJ,N,PUN_End	0.007874016
PRE,N,N_p	0.007874016
N,N_p,PUN_dcot	0.007874016
N_p,PUN_dcot,ADJ	0.007874016
PUN_dcot,ADJ,N_s	0.01181102
ADJ,N_s,N_PRP_s	0.01181102
N_s,N_PRP_s,N_s	0.01181102
N_PRP_s,N_s,AD_s	0.01181102
N_s,AD_s,N_p	0.01181102
AD_s,N_p,V_AUX_Ssm3	0.01181102
N_p,V_AUX_Ssm3,ADJ	0.01181102
V_AUX_Ssm3,ADJ,PUN_dcot	0.01181102
ADJ,PUN_dcot,V_REL	0.01181102
PUN_dcot,V_REL,ADJ	0.01181102
V_REL,ADJ,N_s	0.007874016
ADJ,N_s,V_AUX_Ssm3	0.007874016
N_s,V_AUX_Ssm3,V_IMV	0.007874016
V_AUX_Ssm3,V_IMV,N_sm3	0.007874016
V_IMV,N_sm3,N_PRP_s	0.007874016
N_sm3,N_PRP_s,ADJ	0.007874016
N_PRP_s,ADJ,N_s	0.007874016
ADJ,N_s,ADJ_p3	0.007874016
N_s,ADJ_p3,N	0.007874016
ADJ_p3,N,N_PRP	0.007874016
N,N_PRP,NUM_s	0.007874016
N_PRP,NUM_s,N_s	0.007874016
NUM_s,N_s,ADJ_s	0.007874016
N_s,ADJ_s,ADJ_s	0.007874016
ADJ_s,ADJ_s,N	0.007874016
ADJ_s,N,VIMF_Spm3_O	0.007874016

Appendix G: Geez Script writing alphabet

ሀ	HA	ሁ	HU	ሂ	HI	ሃ	HA	ሄ	HE	ህ	H	ሆ	HO
ለ	LE	ሉ	LU	ሊ	LI	ላ	LA	ሌ	LE	ል	L	ሎ	LO
ሐ	HA	ሑ	HU	ሒ	HI	ሓ	HA	ሔ	HE	ሕ	H	ሖ	HO
መ	ME	ሙ	MU	ሚ	MI	ማ	MA	ሜ	ME	ም	M	ሞ	MO
ሠ	SE	ሡ	SU	ሢ	SI	ሣ	SA	ሤ	SE	ሥ	S	ሦ	SO
ረ	RE	ሩ	RU	ሪ	RI	ራ	RA	ሬ	RE	ር	R	ሮ	RO
ሰ	SE	ሱ	SU	ሲ	SI	ሳ	SA	ሴ	SE	ስ	S	ሶ	SO
ሸ	SHE	ሹ	SHU	ሺ	SHI	ሻ	SHA	ሼ	SHE	ሽ	SH	ሾ	SHO
ቀ	KE	ቁ	KU	ቂ	KI	ቃ	KA	ቄ	KE	ቅ	K	ቆ	KO
በ	BE	ቡ	BU	ቢ	BI	ባ	BA	ቤ	BE	ብ	B	ቦ	BO
ተ	TE	ቱ	TU	ቲ	TI	ታ	TA	ቴ	TE	ት	T	ቶ	TO
ቸ	CHE	ቹ	CHU	ቺ	CHI	ቻ	CHA	ቼ	CHE	ች	CH	ቾ	CHO
ኀ	HA	ኁ	HU	ኂ	HI	ኃ	HA	ኄ	HE	ኅ	H	ኆ	HO
ነ	NE	ኑ	NU	ኒ	NI	ና	NA	ኔ	NE	ን	N	ኖ	NO
ኘ	GNE	ኙ	GNU	ኚ	GNI	ኛ	GNA	ኜ	GNE	ኝ	GN	ኞ	GNO
አ	A	ኡ	U	ኢ	I	ኣ	A	ኤ	E	እ	I	ኦ	O
ከ	KE	ኩ	KU	ኪ	KI	ካ	KA	ኬ	KE	ክ	K	ኮ	KO
ኸ	HE	ኹ	HU	ኺ	HI	ኻ	HA	ኼ	HE	ኽ	H	ኾ	HO
ወ	WE	ዉ	WU	ዊ	WI	ዋ	WA	ዌ	WE	ው	W	ዎ	WO
ዐ	A	ዑ	U	ዒ	I	ዓ	A	ዔ	E	ዕ	I	ዖ	O
ዘ	ZE	ዙ	ZU	ዚ	ZI	ዛ	ZA	ዜ	ZE	ዝ	Z	ዞ	ZO
ዠ	ZHE	ዡ	ZHU	ዢ	ZHI	ዣ	ZHA	ዤ	ZHE	ዥ	ZH	ዦ	ZHO
የ	YE	ዩ	YU	ዪ	YI	ያ	YA	ዬ	YE	ይ	Y	ዮ	YO
ደ	DE	ዱ	DU	ዲ	DI	ዳ	DA	ዴ	DE	ድ	D	ዶ	DO
ጀ	JE	ጁ	JU	ጂ	JI	ጃ	JA	ጄ	GE	ጅ	J	ጆ	JO
ገ	GE	ጉ	GU	ጊ	GI	ጋ	GA	ጌ	TE	ግ	G	ጎ	GO
ጠ	TE	ጡ	TU	ጢ	TI	ጣ	TA	ጤ	CHE	ጥ	T	ጦ	TO
ጨ	CHE	ጩ	CHU	ጪ	CHI	ጫ	CHA	ጬ	PE	ጭ	CH	ጮ	CHO
ጰ	PE	ጱ	PU	ጲ	PI	ጳ	PA	ጴ	TSE	ጵ	P	ጶ	PO
ጸ	TSE	ጹ	TSU	ጺ	TSI	ጻ	TSA	ጼ	TSE	ጽ	TS	ጾ	TSO
ፀ	TSE	ፁ	TSU	ፂ	TSI	ፃ	TSA	ፄ	TSE	ፅ	TS	ፆ	TSO
ፈ	FE	ፉ	FU	ፊ	FI	ፋ	FA	ፌ	FE	ፍ	F	ፎ	FO
ፐ	PE	ፑ	PU	ፒ	PI	ፓ	PA	ፔ	PE	ፕ	P	ፖ	PO

Signed Declaration Sheet

I, the undersigned, 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.

Declared by:

Name: _____

Signature: _____

Date: _____

Confirmed by advisor:

Name: _____

Signature: _____

Date: _____