ADDIS ABABA UNIVERSITY

COLLEGE OF NATURAL AND COMPUTATIONAL SCIENCES

SCHOOL OF INFORMATION SCIENCE

GRAMMAR CHECKER FOR AMHARIC LANGUAGE USING PROBABILISTIC AND RULE-BASED APPROACH

A Thesis Submitted to the School of Information Science of Addis Ababa University in Partial Fulfillment of the Requirement for the Degree of Master of Science In Information Science and Systems (Language Technology)

By Tsedeniya Kinfe

Advisor: Martha Yifiru (PhD)

July 24, 2019

Addis Ababa, Ethiopia

ADDIS ABABA UNIVERSITY

COLLEGE OF NATURAL AND COMPUTATIONAL SCIENCES

SCHOOL OF INFORMATION SCIENCE

GRAMMAR CHECKER FOR AMHARIC LANGUAGE USING PROBABILISTIC AND RULE-BASED APPROACH

By Tsedeniya Kinfe

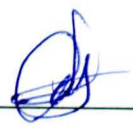
Name and signature of Members of the Examining Board

Martha Yifiru (PhD)	_____	_____
Advisor	Signature	Date
Wondwossen Mulugeta (PhD)	_____	July 30/2019
Examiner	Signature	Date
Dereje Teferi (PhD)	_____	July 30/2019
Examiner	Signature	Date

Declaration

This thesis has not been previously accepted for any degree and is not being concurrently submitted in candidature for any degree in any other university.

I declare that the thesis is a result of my own investigation, except where otherwise stated. I have undertaken the study independently with the guidance and support of my research advisor. Other sources are acknowledged by citations giving explicit references. A list of references is appended.

Signature: 

Tsedeniya Kinfe

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

Advisor's Signature: _____

Martha Yifiru (PhD)

Acknowledgment

All praise and thanks to God for giving me the courage to start and finish my research work. Without His help nothing would have been possible.

Special thanks to my advisor Dr. Martha, for her unreserved guidance, supervision and constructive suggestion during my work. I thank you for encouragement from the beginning till the end of this research work. I would like to thank my friends for their support and encouragement while doing this research.

Last but not list, I would like to thank my beloved family for their understanding, support and unconditional love.

Contents

List of Tables

List of figures

Acronyms

CL	Computational linguistics
DP	Dependency parser
LM	language Model
NLP	Natural language processing
POS	Part of speech
SOV	Subject-Object-Verb
SVO	Subject-Verb-Object
TnT	Trigrams 'n' Tags
WIC	Walta information center

Abstract

Grammar checker is an natural language processing (NLP) application which validates a sentence grammatically based on a predefined rule. Grammar checkers have been developed using different techniques for different languages. Investigation on the development of Amharic grammar checker was conducted by [1] using two approaches (i.e. rule based and statistical approach) independently to handle simple and complex Amharic sentence respectively. The rule-based approach performs better compared to the statistical method in detecting grammatical error for simple sentences. Purpose of this research is, therefore, to investigate the application of a probabilistic method with rule-based approach in the development of Amharic grammar checker. In addition, in the previous study bi-gram and tri-gram LM were used for handling complex sentences. Tri-gram LM achieved better performance but it fails to detect long-distance disagreement within a sentence. Therefore, this paper investigated a long distance agreement by building higher order n-gram LM i.e. 4-gram and 5-gram LM. Moreover, the use of dependency parsing (DP) in Amharic grammar checking has also been investigated.

To conduct the experiment, POS tagged data was used from [2] and this corpus is used for investigating the probabilistic method with rule-based approach and higher order n-gram LM. We have developed automatic POS tagger and chunker to easily identify the subject, object, and verb of a sentence. SRILM toolkit is used to develop tri-, quadra-, and penta-gram LM. The Treebank corpus from [3] is used to investigate dependency parser. To validate and optimize the Treebank MaltOptimizer is used. To train and parse new sentences MaltParser toolkit is used. Each parsed sentence is grammatically analyzed based on the crafted rules. To evaluate the parsed sentence LAS, LA, and UAS metrics are used and are found in MaltEval toolkit.

The result achieved using probabilistic method with rule-based approach is 72% of accuracy while the higher order n-gram method resulted in 65%, 67% and 65.6% for tri-, quadra-, penta-gram language models, respectively. Whereas dependency parser scores 81.5%, 94.2% and 84.4% in LAS, UAS, and LA respectively. The overall accuracy achieved by DP is 84.7% in detecting grammatical errors.

Chapter One: Introduction

Natural Language Processing (NLP) is a means for computers to understand and manipulate human language and produce a meaningful output. It is an automatic process of a computer extracting meaningful information from natural language input and/or producing natural language output. Human language technology covers a broad range of activities consisting of Computational Linguistics (CL) and speech technology as its core. CL is an interdisciplinary field dealing with the statistical and/or rule-based modeling of natural language from a computational perspective[4].

CL also deals with syntax, which is the study of structural relationships between words in a certain language. The relationship among words is governed by a set of rules which are used to structure linguistic expressions [5]. Any natural languages have different sentence structure. Sentence structure depends on the position of subject, object, and verb. For instance, English has an SVO structure whereas Amharic has SOV structure.

Grammar (or syntax) refers to a system of rules describing what correct sentences should look like [5]. Parser and grammar can sometimes be used interchangeably hence parser identifies the structure of sentences in which sentences are made up of information units (namely, noun phrase, verb phrase). And therefore it also deals with a number of sub problems such as identifying constituents that can fit together, testing the compatibility of number, gender, person and/ or tense[6]. This is the task of grammar checker application.

A grammar checker is an NLP application that validates a sentence syntactically and provides feedback with an explanation of the mistake and suggests how to correct it. A grammatical error can be caused by typographical error or by lack of knowledge in the language. Typographical (error) is an error in typing or printing especially caused by striking an incorrect key on the keyboard, for example unintentionally entering wrong determiner or adding a plural marker at the end of the word. There are two types of grammatical errors: - structural errors those which can only be corrected by inserting, deleting, or moving one or more words and nonstructural errors are those which can be corrected by replacing an existing word with a different one or by word reordering [7]. A missing word can be a real problem in grammar checker because grammar does not have semantic knowledge – it does not understand the meaning of the word

and it is very difficult to determine a missing word and what it might be [8],[9]. However, there are methods for handling structural and nonstructural errors by using different approaches.

Grammar checking can be implemented in three approaches: statistical, rule-based, and syntax-based. The statistical/ corpus-based approach utilizes a POS tagged annotated corpus to build a list of POS tag sequence. If the sequence of POS does exist then the input text is considered as grammatically correct if not, incorrect. In the rule-based approach, rules are manually crafted. A set of rules are matched against a text. The last approach is a syntax-based approach which is a technique that uses a text which is completely parsed, i.e. the sentences are analyzed and each sentence is assigned a tree structure. The text is considered incorrect if the parsing does not succeed [10]. The statistical and rule-based approach can be applied by integrating the two methods creating **hybrid approach** which provides robustness and efficiency [11].

1.1. Statement of the problem

Grammar Checker is important to validate sentences and avoid grammatical mistakes. Grammar checker has been developed for foreign language for instance English [8], Swedish [12], Punjabi [13], Bangla [14] using different tools and technique. But for our local language, there are only two researches that have been conducted. These are grammar checker for Afan-Oromo [15]and Amharic language [1].

Grammar checker for the Amharic language was done by [1] in 2013 using two approaches i.e. rule-based and statistical approaches. The experiment is conducted by classifying sentences in to two groups, simple and complex sentence. A simple sentence is a sentence which has only one verb phrase. Whereas a complex sentence is a sentence which is categorized as simple sentence and formed by complex phrases [16]. The rule based approach was implemented and tested on Amharic simple sentences. Statistical method was implemented for simple and complex sentence by building a bi-gram and tri-gram language models (LM).

For the simple sentences, rule based approach was compared against bi-gram and tri-gram LM. For the complex sentence bi-gram LM was compared with tri-gram LM. The findings showed that, for complex sentence better performance was achieved by tri-gram LM. Rule-based approach surpasses both bi-gram and tri-gram LM on the simple sentences.

Therefore, the current research investigated the use of statistical (probabilistic) method with rule based approach for the Amharic language grammar checker for both simple and complex sentences. In addition, since the previous research considers a window size of three to check the grammatical correctness of complex sentences, the system fails to detect long-distance grammatical correctness of Amharic sentence. Thus, we investigated the use of higher order n-gram language model. Moreover, effect of dependency parsing in long distance grammatical agreement has also been investigated. Additionally dependency parsing was investigated on handling a slight free word behavior of the language. The previous researcher takes only SOV word order as a correct word sequence of the language even though OSV can convey meaningful sentence.

Phrasal structure or constitutional parser was developed by many researchers for Amharic language by applying different techniques (Automatic Sentence parsing for Amharic text by using Probabilistic Context Free Grammar (PCFG) [17] An Integrated Approach to Automatic Complex Sentence Parsing for Amharic Text [18], base phrase chunker and parsing [19] and Top down chart parsing for Amharic sentences[6]. These researches showed the sub–phrasal relation of Amharic text with in a sentence. The interest of this work is, however, to identify the relation between words in a sentence so as to come up with features that best help parsers learn the syntax and choose among different parses.

1.2. Research question

The research answers the following questions.

1. Which of the chosen approach (higher order n-gram and DP) is suitable for handling long distance agreement and free word order?
2. How does the use of probabilistic method with rule- based approach improve the performance of Amharic grammar checker?

1.3. Objective

1.3.1. General objectives

The general objective of the research is to investigate the use of probabilistic method with rule-based approach in the development of Amharic grammar checker and examine the use of higher order n-gram and dependency parsing in handling long distance and free word order grammatical agreement in Amharic sentence respectively.

1.3.2. Specific objectives

To achieve the general objective of the research, the following specific objectives are set out.

- To study related research papers on the domain area.
- To study the syntactical/ grammatical behavior of Amharic language.
- To construct tagged corpus of number, gender and person feature marker.
- To explore the appropriate model for the Amharic grammar checking system.
- To investigate a POS tag with syntax feature marker (number tag, person tag and gender tag) language model.
- To evaluate the performance of the designed algorithm and model.

1.4. The significance of the study

Grammar checker helps Amharic writers to write a grammatically correct text. It helps to write an error-free collection of documents. It also contributes to other application which needs grammar checker as a sub-component of their application, word processor can be mentioned as example.

Considering the purpose of the research work the following can be considered as a major contribution of the study.

- Grammar checker for the Amharic language by using probability method with rule-based was conducted. The result of our experiment shows that this approach is suitable for developing Amharic grammar checker.
- A higher order n-gram LM (3-gram, 4-gram and 5-gram LM) for handling long distance agreement in a sentence is investigated. 5-gram LM did not show a significant increase in error detection than 3-gram and 4-gram LM due to sparse data. Better result was achieved by tri-gram LM.
- DP is conducted for identifying grammatically in/valid sentence. This experiment is conducted to handle the long distance dependency in Amharic sentence. As the result of the experiment shows DP is suitable for handling long distance agreement of Amharic language. Amharic language has SOV word order but there are cases when object comes before the subject of the sentence. As of the finding, for the slightly free word order behavior of the language dependency parsing is suitable to handle the relation of the word within a sentence.

1.5. The scope and limitation of the study

This research paper is confined to study and analysis of detecting common grammatical errors. Errors in number, gender, person and tense within a sentence are covered in this work. However, disagreements that occur beyond the sentence boundary is not incorporated. The system detects and analyzes adjective- subject, subject- verb, adverb- verb and object- verb agreement in Amharic text.

The other limitation of the study is the system cannot handle if the input word (e.g. verb) has same person for the subject and object. For example '*መጥቼልኝ*'/ meTcElN/ indicates the benefactive marker of the subject and object is the same person i.e. 'first person'. This kind of invalid sentence can be detected by spell checker but spell checker only used as preprocessing in this thesis.

User must input a sentence which is annotated by morpho- syntactic feature to analyze the input sentence using higher n-gram language model therefore this approach only accepts a sentence with their grammatical feature.

1.6. Organization of thesis

This chapter gives a brief background of grammar checker and presents objective, scope of the study and problem that are addressed in this research. Then a review of literature that discusses various theories and concepts on grammar checker is presented in chapter two and related research work on different language is presented in chapter three. The next chapter i.e. chapter four presents the research methodology. In chapter five evaluation and analysis of common grammatical error are discussed. Conclusion and recommendation for the future work is presented in chapter six.

Chapter Two: Literature Review

This section introduces the fundamental aspect of this work by first discussing what grammar and grammar checker is. Following this, methods or approaches that are used to develop a grammar checker has been discussed. And finally, the behavior and syntax of Amharic language is presented.

2.1. Definition of grammar

People communicate and share information, knowledge, ideas, and thoughts using natural language. To complete communication and to make it meaningful, a language must follow a set of rules. Grammar studies a significant element in the language and a set of rules that make it coherent[11]. Grammar is the syntax of a language which describes how words are combined to form a sentence.

A grammar of a natural language is a set of combination (syntax) and modification (morphology) of components and words of the language to form a sentence[7]. The process of verification of syntax and morphology is called grammar checking and the tool that performs this task is called grammar checker[7].

According to [20] grammar is defined as "the branch of linguistic science which is concerned with the description, analysis, and formalization of formal language patterns." Rules are the sequence of POS tags, agreement and word order in a given language. Therefore, POS tagging plays a significant role in building grammar checker.

2.2. Approach to develop grammar checker

2.2.1. Syntax-based approach

In this approach, a text is completely analyzed morphologically and syntactically. The parser assigns a syntactic structure to each sentence. A text is completely parsed, i.e. the sentences are analyzed and each sentence is assigned a tree structure. The text is considered incorrect if the parsing does not succeed [8]. According to the level of the linguistic analysis to which the error belongs, syntax-based checking can be classified as either a deep syntactic analysis or a shallow syntactic analysis [21].

Shallow syntactic processing designates methods for generating partial analysis of sentences and includes tools such as chunkers (dividing sentences into chunks) and parts-of-speech taggers (assigning a syntactic label such as NP to a chunk) [22]. It is used to extract syntactic relation or patterns mainly by using regular expression to be fetched over a text or a chunked text. There are only limited possibilities to extract reliable relations using only shallow syntactic parsing. Based on regular expressions, it is possible to extract some patterns such as NP VP NP. But this is far from being sufficient and this does not cope with language ambiguities such as long-distance relationships. Therefore, shallow parsing shows its limits at this level [22].

Deep syntactic parsing is performed using a set of grammar rules that assign parse trees to sentences. The parser learns knowledge about a language using hand- labeled sentences and produces analyses when parsing sentences. The output representations can take the form of phrase structure tree representations or dependency parses. Phrase structure parses associates a syntactic parse in the form of a tree to a sentence, while dependency parses creates grammatical links between each pair of words in the sentence[22].

The advantage of the syntax-based approach is that off-the-shelf NLP resources such as lexicons, morphological analyzers and parsers can be used to do the analysis. Unfortunately, the checker will only recognize that the sentence is incorrect, it will not be able to tell the user what the exact problem is. For this, extra rules are necessary in order to either parse ill-formed sentences or apply a technique to features associated with linguistic fragments.

Top down parsing

The basic idea is to start at the root node, expand the tree by matching the left hand side of rules. And derive a tree whose leaves match the input. It is a strategy of analyzing unknown data relationships by hypothesizing general parse tree structures and then considering whether the known fundamental structures are compatible with the hypothesis. It can be viewed as an attempt to find left most derivations of an input by searching for parse-trees using top-down expansion of the given grammar rules.

A top down parser will derive a tree in a succession of stages, starting with just a single node. At every stage of the process of tree derivation there are choices to be made. One is concerned about which node to expand and the other might be concerned about how to expand each node.

Bottom up parsing

The basic idea is to start with the leaves, build a tree by matching the right hand side of rules then build a tree with S at the root. It is data driven in which it attempts to parse the words that are there. It looks at words in input string first, then checks their categories and tries to combine them into acceptable structures in the grammar. It also involves scanning the derivation for sub-strings which match the right hand side of grammar rules and using a rule that would show their derivation from the non-terminal symbol of the rule. Bottom up parsing might be inefficient when there is great lexical ambiguity.

2.2.2. Dependency Parsing

Dependency parsing is the task of extracting a dependency parse of a sentence that represents its grammatical structure and defines the relationships between head words and words, which modify those heads.

Transition based dependency parsing

Transition based parsing is motivated by a stack based approach called shift-reduce parsing which was developed for analysis of programming language. This approach is simple that employs a context free grammar, a stack and list of tokens to be parsed. The approach is adapted and actions are replaced and used for the transition based approach.

The parser starts in an initial configuration. At each step, it asks a guide to choose between one of several transitions into new configurations. Parsing stops if the parser reaches a terminal configuration. The parser returns the dependency tree associated with the terminal configuration.

A key element in transition based parsing is the notion of a configuration which consists of a stack, input words or tokens and set of relations representing a dependency tree. Given this, the parsing process consists of a sequence of transitions through the space of possible configurations. The goal of this process is to find an appropriate dependency tree that is synthesized.

In standard approach to transition-based parsing, the operators used to produce new configurations are simple and correspond to the intuitive actions taken to create dependency tree by examining the input words from left to right. The transition operators operate on the top two elements of the stack.

Graph based dependency parsing

Graph based approaches to dependency parsing search through the space of possible trees for a given sentence for a tree that maximize some score. These methods encode the search space as directed graphs and employ methods drawn from graph theory to search the space for optimal solutions.

There are several motivations for the use of graph based methods. The first are capable of producing non-projective trees. The second motivation is concerned about parsing accuracy specifically with respect to longer dependencies. Empirically transition based methods have high accuracy on shorter dependency relations but accuracy is reduced as the distance between the head and dependent increases. Graph based methods have come up with scoring entire trees to avoid this difficulty.

2.2.3. Parsing algorithm

There are different Parsing algorithms Of MaltParser having mainly three families. The first one is Nivre's algorithms that include the arc-eager and arc-standard versions of the algorithm described in Nivre[23]. The second one is Covington's algorithms that include the projective and a non-projective version of the algorithm. The third algorithm is the Stack algorithms which deal with the projective and non-projective versions of the algorithm. While both the Covington and the Stack family contains algorithms that can handle non-projective trees, the Nivre's family does not. If there are no non-projective dependencies in the training set and when only projective algorithms are explored. If the training set contains a substantial amount of non-projective dependencies, then MaltOptimizer instead tests Covington's non-projective algorithm and the non-projective Stack algorithms as well as projective algorithms in combination with pseudo-projective parsing.

The training data is projectivized by a minimal transformation, lifting non-projective arcs one step at a time, and extending the arc label of lifted arcs using the encoding scheme called HEAD, which means that a lifted arc is assigned the label $r \wedge h$, where r is the original label and h is the label of the original head in the non-projective dependency graph. MaltParser also provides an option for a non-projective transition system based on the method as described above in Covington algorithm. This system uses a similar type of configuration of arc-eager but adds a second temporary stack. Unlike the arc-eager, this allows the derivation of arbitrary non projective dependency trees [23].

We can apply both group algorithms if the amount of non-projective dependencies is in-between. Minimally, the default algorithm nivre eager should be compared to stackproj, since one of these two algorithms often performs best in most cases. Some of the parsing algorithms have options of their own that can be tuned [24]

2.2.3.1. Niver's algorithm

It is a linear-time algorithm limited to projective dependency structures. It is incremental as it processes the tokens in the order they appear in the sentence. This means that we can start parsing while the input is still being produced, for instance in a spoken dialogue system or input system for mobile phones. There are four actions in the Niver's algorithm: shift, reduce, left-arc and right-arc.

The Nivre algorithm can be run in arc-eager (-a E) or arc-standard (-a S) mode. In addition allow_root and allow_reduce options are available. If **allow_root=true**, the parser treats the special root node as a token during parsing, allowing root dependents to be attached with a RightArc transition; otherwise root dependents are not attached during parsing. In both cases, unattached tokens are attached to the special root node with the default label after parsing is completed. If **allow_reduce=true**, the Reduce transition is permissible even if the node on top of the stack does not have a head. As a result, this node will be attached to the special root node after parsing is completed, which may give rise to non-projective trees.

The allow_root option decides whether the parser will start parsing with an artificial root token on the stack (true) or with an empty stack (false), and the allow_reduce option decides whether

the reduce transition is allowed even if the token on top of the stack does not have a head (true) or whether only attached tokens can be reduced (false). The enforce_tree option can be used with the arc-eager version to make sure that the output parse is a tree.

Nivre's algorithm uses two data structures:

- A stack **Stack** of partially processed tokens, where **Stack[i]** is the i+1th token from the top of the stack, with the top being **Stack[0]**.
- A list **Input** of remaining input tokens, where **Input[i]** is the i+1th token in the list, with the first token being **Input[0]**.

2.2.3.2. Covington's algorithm

It is a quadratic-time algorithm for unrestricted dependency structures, which proceeds by trying to link each new token to each preceding token. It can be run in a projective (-g P) mode, where the linking operation is restricted to projective dependency structure or in non-projective mode, allowing non- projective dependency structures. In non-projective mode, the algorithm uses the context stack to store unattached tokens occurring between the stack and input. The algorithm work one word at a time and attempt to build a connected dependency graph with only a single left-to-right pass through the input.

There are two options, allow shift and allow root that controls the behavior of Covington's algorithm. If **allow_shift=true**, Shift is a valid transition, allowing the parser to skip remaining tokens in Left; otherwise all tokens in Left must be inspected before the next token is shifted. If **allow_root=true**, the parser treats the special root node as a token during parsing, allowing root dependents to be attached with a RightArc transition; otherwise root dependents are not attached during parsing. In both cases, unattached tokens are attached to the special root node with the default label after parsing is completed.

Covington's algorithm uses four data structures:

- A list **Left** of partially processed tokens, where **Left[i]** is the i+1th token in the list, with the first token being **Left[0]**.

- A list **Right** of remaining input tokens, where **Right[i]** is the i+1th token in the list, with the first token being **Right[0]**.

- A list **LeftContext** of unattached tokens to the left of **Right[0]** (and to the right of **Left[0]**), where **LeftContext[i]** is the i+1th such token, with **LeftContext[0]**being the token immediately to the left of **Right[0]**.

- A list **RightContext** of unattached tokens to the right of **Left[0]** (and to the left of **Right[0]**), where **RightContext[i]** is the i+1th such token, with **RightContext[0]** being the token immediately to the right of **Left[0]**.

2.2.3.3. Stack algorithm

An input sentence is parsed from left to right. The parser has a stack and there are two basic actions on the stack: shift and reduce. The shift action pushes the next word in the input sentence on to the top of the stack.

The Stack algorithms are similar to Nivre's algorithm in that they use a stack and a buffer but differ in that they add arcs between the two top nodes on the stack (rather than the top node on the stack and the first node in the buffer) and that they guarantee that the output is a tree without post-processing.

The Projective (**-a stackproj**) Stack algorithm uses essentially the same transitions as the arc-standard version of Nivre's algorithm and is limited to projective dependency trees. The Eager (**-a stackeager**) and Lazy (**-a stacklazy**) Stack algorithms in addition make use of a swap transition, which makes it possible to derive arbitrary non-projective dependency trees. The Eager algorithm applies the swap transition as soon as possible, while the Lazy algorithm postpones swapping as long as possible.

The Stack algorithms use three data structures:

- A stack **Stack** of partially processed tokens, where **Stack[i]** is the i+1th token from the top of the stack, with the top being **Stack[0]**.

- A list **Input**, which is a prefix of the buffer containing all nodes that have been on Stack, where **Input[i]** is the i+1th token from the start of **Input**.

- A list **Lookahead**, which is a suffix of the buffer containing all nodes that have **not** been on Stack, where **Lookahead[i]** is the i+1th token from the start of **Lookahead**.

Note that it is only the swap transition that can move nodes from **Stack** back to the buffer, which means that for the Projective Stack algorithm **Input** will always be empty and **Lookahead** will always contain all the nodes in the buffer.

2.2.4. MaltParser (A Transition-based Dependency Parser)

We use a data-driven dependency parser 'MaltParser' for our research. MaltParser [45] implements the transition-based approach to dependency parsing, which has two essential components.

- A transition system for mapping sentences to dependency trees

- A classifier for predicting the next transition for every possible system configuration.

Given these two components, dependency parsing can be realized as deterministic search through the transition system, guided by the classifier. With this technique, parsing can be performed in linear time for projective dependency trees and quadratic time for arbitrary[45].

2.2.5. MaltEval: An Evaluation and Visualization Tool for Dependency Parsing

MaltEval, an evaluation software for dependency parsing developed by [25], and is freely available. It has been created to make evaluation and visualization of dependency structure easier. The currently available values for Metric are shown below, where two different values can be used for the first three:

LAS (BothRight) A token is counted as a hit if both the head and the dependency label are the same as in the gold-standard data. This is the default value.

LA (LabelRight) A token is counted as a hit if the dependency label is the same as in the gold-standard data.

UAS (HeadRight) A token is counted as a hit if the head is the same as in the gold-standard data.

AnyRight A token is counted as a hit if either the head or the dependency label (or both) is the same as in the gold-standard data.

BothWrong A token is counted as a hit if neither the head nor the dependency label are the same as in the gold-standard data.

LabelWrong A token is counted as a hit if the dependency label is **not** the same as in the gold-standard data.

HeadWrong A token is counted as a hit if the head is **not** the same as in the gold-standard data.

AnyWrong A token is counted as a hit if either the head or the dependency label (or both) is **not** the same as in the gold-standard data.

2.2.6. Statistical based approach

This approach uses a large corpus which covers language features and the linguistic phenomenon of the language. Under this approach, two sub-approaches are involved. The first, corpus-based approach ensures correctness of sentence by checking the input text against the corpus. The second approach deals with probabilistic/statistical checking of input text[7]. In the statistical approach, the training corpus is POS tagged text. If the sequence of Parts of Speech (POS) tag is very common then the text will be considered as correct and if the sequence did not occur at all or is rare, the input text will be considered as incorrect hence POS plays vital role in probabilistic/statistical approach. POS and POS tagger is further discussed in 2.2.7. N-gram is one of type of a statistical based approach, which is probabilistic language modeling for predicting the next item in a given sequence [26]. Building the n-gram LM allows us to assess naturalness of a sentence and to generate a text. Given a target sentence, the LM can tell us whether the sentence is an actual natural language in the target language; we can use the language model to check the fluency of a sentence generated by humans for the purpose of grammar checking or error correction [27].

N-gram probability can be calculated by preparing a set of training data from which we can count strings (it can be letters or characters, words, symbols, tags which can be POS), count up the number of times we have seen a particular string and divide it by the number of times we have seen in the context.

Based on N value, different names can be given for the sequences. Bi-gram, tri-gram, quadra-gram, and penta-gram are for 2, 3, 4 and 5 values of N respectively. The higher order N-gram is used when we want to comprehend more context information [27]. For this reason, we used higher order n-gram to handle long distance grammatical agreement within Amharic sentence by building n-gram of tags, a sequence of tags that describes such a subsequence of neighbored tokens in a sentence, where n defines the number of tokens [7].

2.2.7. POS tagger

POS is a linguistic category of a word based on a predefined word class [28]. POS can be categorized as an open and closed word class. Open word classes are those which convey a true lexical meaning of the sentence like noun, verbs, and adjectives. The closed word class has a small number of classes to define lexical categories like prepositions, postposition, determiners etc [27].

POS can tell us which morphological affixation it can take to enhance an application (for instance information retrieval, grammar checker) by selecting the important word from the document [27].

The main task of POS tagger is to assign the appropriate word class and morpho-syntactic feature to each token in a text. This is used in many NLP tasks like Machine Translations, Information Extraction, Grammar checker, Parsing etc. In most of the cases, the accuracy of these NLP applications depends upon the accuracy of POS tagger.

POS tagging can be implemented by using large number of hand-crafted rules or by training using large amount of corpus i.e. probabilistic tagger. HMM and TnT can be mentioned as one of probabilistic tagger and both algorithms are used in our research, detail about the algorithms as follows.

HMM

A hidden Markov model (HMM) is a finite state automaton with stochastic state transitions and observations. The automaton models a probabilistic generative process whereby a sequence of observations is produced by starting in some state, emitting an observation selected by that state, passing to a new state, emitting another observation-and so on until a designated destination state is reached[27].

More formally, an HMM model is characterized by the following [29].

N is the number of states in the model. Generally, the states are interconnected in such a way that any state can be reached from any other state. The hidden states often represent important physical aspects. In POS tagging, the states represent the tags. We denote the individual states as T = t_1,t_2, ...t_n, and the state at time t as q_n .

V, the number of distinct observation symbols per state, i.e., the discrete alphabet size. The observation symbols correspond to the physical output of the system being modeled. For POS tagging, the observation symbols are the words of a given language. We denote the individual symbols as W = w_1, w_2.....w_n.

The state transition probability distribution A = $\{a_{ij}\}$ where

$$a_{ij} = P\left(q_{t+1} = \frac{t_j}{q_t} = t_i\right),\ 1 \leq i, j \geq N,\ a_{ij} \geq 0$$

When there are no connections between states, the corresponding a_{ij} transition probability is zero. For all other cases, it is greater than zero.

The observation symbol probability distribution in state j, B = $\{b_k\}$ Where

$$b_k = P\left(\frac{w_k}{q_t} = t_j\right),\ 1 \leq i, j \geq N,\ 1 \leq k \geq V$$

The initial state distribution $\pi = \{\pi_{i\}}$, where

$$\pi_I = P(q_1 = t_i), 1 \leq i \geq N,$$

The above enumeration shows that HMM models have two parameters and three probability distributions. The two parameters are N (number of states) and V (vocabulary size) and the three probability distributions are A, B, and π. The three probability distributions are henceforth referred to as λ where

$$\lambda = (A, B, \pi)$$

Given an HMM model with values for N, V, and π, there are three problems that are of interest as formulated by[29]:

> **Problem 1** Given the observation sequence $O = O_1, O_2 \ldots O_{tt}$ and a model$\lambda = (A, B, \pi)$, how do we efficiently compute P(O/λ)? In other words, what is the probability of the observation sequence, given the model?
>
> **Problem 2** Given the observation sequence $O = O_1, O_2 \ldots O_t$ and a model$\lambda = (A, B, \pi)$, how do we choose the underlying state sequence $q = q_1, q_2 \ldots q_t$?
>
> **Problem 3** How do we adjust the probability distributions $\lambda = (A, B, \pi)$to maximize $P(O/\lambda)$?

Such are the questions that can be raised about an HMM model. Solving the first problem is important in comparing two or models. We choose a model that gives high probability to observation sequences. Solution to the second problem is important in discovering the underlying hidden state sequences that could have given rise to the observation. The solution to the third problem has the advantage of giving us the best model for the observations.

Usually, the relevant problem for grammar checking is the second problem i.e. discovering the underlying tag sequences for a given sequence of words.

More formally, given a sequence of words $W = w_1, w_2 \ldots\ldots w_I \ldots w_N$ where $w_I \Sigma V$ (Vocabulary), what is the most probable sequence of tags $Tt_1, t_2 \ldots\ldots t_I \ldots t_N$ where $w_I \Sigma V$ (Tagset) that could have given rise to these words?

To solve this problem, we need to have appropriate values for the model $\lambda = (A, B, \pi)$ and an efficient algorithm to search through the space of tag sequences for the optimal one. The values

for λ can be estimated from a tagged corpus. Estimating the value for $\lambda = (A, B, \pi)$ can be varied. The Viterbi algorithm, a dynamic programming based algorithm, is used to find the best sequence [27], which basically involves solving the following probability.

$$T = arg_{Tmax} P\left(\frac{T}{W}\right)$$

$$arg_{Tmax} P(T)$$

$P(T)$and $P(W/T)$ are the transition probabilities and observation(lexical) probabilities, respectively. arg_{max} tells us that the function returns the tag sequence that maximizes the probability function value. Transition probability captures more of context and tag dependencies.

$$P(T) = P(t_1, t_2 \ldots\ldots t_n)$$

$P(t_1)P(t_2/t_1)P(t_3/t_2\, t_1)::: P(t_n/t_{n-1}\, t_{n-2} \ldots\ldots t_1)$.....Chain Rule (3.2)

This equation assumes that the current tag depends on all previous tags. It is hard to find values for such an assumption. The common way to do is to assume the current tag depends on some fixed number of previous tags. Depending on this fixed number, we have unigram, bi-gram, and N-gram in general. Unigram means the current tag does not depend on previous tags. Bi-gram means the current tag depends on the previous tag and so on. For example, for the bi-gram case, the equation becomes as shown below.

Transition probability:

$$P(T) = P(t_1)P\left(\frac{t_2}{t_1}\right)$$

Lexical probability:

$$P\left(\frac{w_1}{t_1}\right) P\left(\frac{w_2}{t_2}\right)\left(\frac{w_3}{t_3}\right) \ldots\ldots\ldots \left(\frac{w_n}{t_n}\right)$$

$$\prod_{i=1}^{n} P\left(\frac{w_i}{t_i}\right)$$

This equation assumes the current word is determined completely by its tag. The values for the transition and lexical probabilities are estimated by the maximum likelihood estimate given below for the bi-gram case.

$$P\left(\frac{t_i}{t_{i-1}}\right) = \frac{Count(t_{i-1},t_i)}{Count(t_{i-1})} \quad \textit{A parameter}$$

$$P\left(\frac{w_i}{t_i}\right) = \frac{Count(t_i,w_i)}{Count(t_i)} \quad \textit{B parameter}$$

Using the above two equations, A and B parameters can easily be calculated by counting and dividing. In counting, it is possible that for some sequences of tokens may get zero. This can be bad because it implies particular tokens can never occur. However, this is not necessarily true. It may just mean that the training corpus did not have the tokens. To deal with this count zero problem smoothing techniques can be applied.

TnT algorithm

TnT, the short form of *Tri-grams 'n' Tags*, is a very efficient statistical POS tagger that is trainable on different languages and virtually any tag set [30]. The component for parameter generation trains on tagged corpora. The system incorporates several methods of smoothing and of handling unknown words.

The tagger is an implementation of the Viterbi algorithm for second orders Markov models. The main paradigm used for smoothing is a linear interpolation; the respective weights are determined by deleted interpolation hence there is no discounting in TnT! [31]

Interpolation-based smoothing:

$P(t_3/t_1,t_2) = \lambda_1 P(t_3) * \lambda_2 P(t_2/t_1) * \lambda_3 P(t_3/t_1,t_2)$,where t_1, t_2, t_3 is a tag set and$\lambda_1 + \lambda_2 + \lambda_3 = 1$.The λ coefficients are also estimated from the training data (deleted interpolation)

Unknown words are handled by a suffix trie and successive abstraction. [31]discussed unknown words can be handled by an n-gram model over letters. It also models capitalization and has an

efficient decoding algorithm (beam-search pruned Viterbi). Unknown words are calculated using a letter-based n-gram, using the last m letters li of an L-letter word:

$$P(t|lL-m+1,\ldots,lL)$$

[31] indicates the basic idea for handling unknown words are by using suffixes of unknown words which give a good clue to the POS of the word. But how big is m? m should not be bigger than 10, but it is based on the longest suffix found in the training set. These probabilities also smoothed by interpolation.

TnT is not optimized for a particular language. Instead, it is optimized for training on a large variety of corpora. Adapting the tagger to a new language, new domain, or a new tag set is very easy. Additionally, TnT is optimized for speed [30].

2.2.8. Rule-based approach

This approach is a more common way to do grammar checking [7]. Rules are generated manually. Rules are easy to configure and can be easily added, removed or updated. But if sentences are more complex, the rules to detect errors will also be complicated[32]. The most significant advantages include, rules can be handled by one who does not have programming knowledge like linguistics and it provides a detailed error message. The main feature of this method is that it provides all feature of a language moreover sentences also do not need to be complete to be checked against the rules i.e. it can handle input text while writing. [12],[33],[15], [8] are few of researches that were candidate using rule-based approach.

Pure empirical approach is advisable way to create rules which rely on error corpora, these are hard to find and costly. Although it is possible to automatically bootstrap the development of such corpora [34].

Pure prior approach is based on existing normative prescription. Such as dictionary, grammar text books, style guidance. This approach has an advantage over the empirical approach i.e. rules seems to be more reliable and it is easier to find standard format and use the corpus to find what derivations seems to occur most frequently [34]. It is also possible to use machine learning

algorithms with some existing language resource such as transformation-based learning, statistical machine translation.

2.3. The Amharic language

Amharic is a language spoken in Ethiopia which is classified in the Semitic language family [35]. Semitic languages are characterized by a productive system of more standard affixation processes. These include prefixes, suffixes, infixes and circumfixes, which are involved in both inflectional and derivational processes [36]. The Amharic language is morphologically rich language, for instance, if we take verbs of Amharic language, they are inflected in person, gender, number, aspect, tense, and mood of the sentence. A single verb in Amharic can also be a complete sentence; it can hold both the subject and object within the verb itself. For example let's consider 'አልነገረኝም'/' *'alnegereNm'* in Amharic, which means 'he didn't tell me' in English, specifies the subject (he) and the object (me) in a word.

2.3.1. Word formation

Amharic morphology is complex, particularly verbs [2]. Amharic noun and adjectives are inflected for case, number, gender and person whereas verbs are inflected for person, number, gender, tense-aspect-mood (TAM) and case. Verbs may also contain pronoun object marker.

2.3.1.1. Inflectional behavior

Verb

Verbs are inflected for person, gender, number, aspect, tense, and mood[35]. For indicating a person, gender and number suffix are added to the stem. [37] Illustrates the inflectional behavior of Amharic verb as shown in the Table 1.

1st Person singular	**1st person plural**	
ሰበር- ኩ	ሰበር-ን	
2nd person masculine	**2nd person feminine**	**Plural**

ሰበር-ኀ	ሰበር-ሳ	ሰበር-አችው
3rd person masculine	3rd person feminine	Plural
ሰበር-ኦ	ሰበር-ኦት	ሰበር-ኡ
Polite 2nd& 3rd		
ሰበር-ኡ		

Table Inflectional behavior of Amharic verbs (definiteness marker)

Tense has a different format, present tense in the Amharic language is indicated using the verbs "ነው ". For past tenses "ነበር" is used. For continuous tense form "እየ-" prefix is used. If we insert the verb "ነው " or "ነበር " to the end of Amharic sentence along with continuous form indicator, the form will have present continuous or past continuous tense form respectively.

The other inflectional behavior is indicated by benefactive and malfactive indicators. See table 2 below [37].

		Benefactive	Malfactive
1st person	Singular	ሰበር- ኧልኝ	ሰበር-ኧብኝ
	Plural	ሰበር- ልን	ሰበር-ብን
2nd person	Masculine	ሰበር- ልህ	ሰበር- ብህ
	Feminine	ሰበር- ልሽ	ሰበር- ብሽ
	Plural	ሰበር- ላችሁ	ሰበር- ባችሁ
	Polite	ሰበር- ኦሎት	ሰበር-ኦቦት
3rd person	Masculine	ሰበር- ኧላት	ሰበር- ኧባት
	Feminine	ሰበር- ኧላት	ሰበር- ኧባት

Plural	ሰበረ-ላቸው		ሰበረ-ባቸው
Polite	ሰበረ- ላቸው		ሰበረ- ባቸው

Table 1 Inflectional behavior of Amharic verbs (benefactive and malfactive)

There are four moods in Amharic: declarative, interrogative, negative and imperative. Verbs can take different forms according to the mood [37].

		Declarative	Interrogative	Negative	Imperaltive
1st person	Singular	ሰበረ	ል- ሰበር	አል- አ-ሰበር	
	Plural	-ን- ሰበር	-ን- ሰበር	አል- ን-ሰበር	
2nd person	Masculine	ት- ሰብር	ት- ሰብር	አት-ሰበር	ሰበር
	Feminine	ት- ሰበር- ኢ	ት- ሰበር- ኢ	አት- ሰበር- ኢ	ሰበር- ኢ
	Plural	ት- ሰብር- ኡ	ት- ሰብር- ኡ	አት- ሰበር-- ኡ	ሰበር- ኡ
	Polite	ት- ሰብር- ኡ	ት- ሰብር- ኡ	አት- ሰበር-- ኡ	ሰበር- ኡ
3rd person	Masculine	ይ- ሰበር	ይ- ሰብር	አይ- ሰብር	ይ- ሰብር
	Feminine	ት- ሰበር	ት- ሰብር	አት- ሰብር	ት- ሰብር
	Plural	ይ- ሰበር- ኡ	ይ- ሰብር- ኡ	አይ- ሰብር- ኡ	ይ- ሰብር- ኡ
	Polite	ይ- ሰበር- ኡ	ይ- ሰብር- ኡ	አይ- ሰብር- ኡ	ይ- ሰብር- ኡ

Table Inflectional behavior of Amharic verbs (mode)

Noun

Amharic nouns can be marked for numbers, definiteness, gender, and case [35]. Noun number markers are "-ኦች" suffix for the noun which ends with a consonant and"-ዎች" suffix is used to the noun with vowel ending. For indicating personal pronoun and proper noun "እነ-" prefix is used.

The plural formation can be formed by repetition, for instance, the plural form of "ቅጠል" will be "ቅጠላቅጠል" and the borrowed nouns from geeze is formed by "-አች","-አን" or "-አት" morphemes [16].

Definiteness indicated by affixation of morphemes or vowels based on the number, gender, and/or ending of the noun. A morpheme that indicates a singular masculine is"-ኡ" suffix and singular feminine are indicated by using "-ዋ" or "-ኢቱ" suffixes. The plural marker is "-አች-ኡ". The above morphemes are used for the nouns with consonant endings. On the other hand, the noun which has vowel ending will have "-ው" suffix for singular masculine indicator and for singular feminine "-ዋ" or "-ይቱ" is used. For the plural indicator "-ዎች-ኡ" is used.

Case indicator can be an objective case by using "-ን" or possessive case by affixation of morphemes or vowels based on the person, number, gender, and/or ending of the noun (personal pronouns by prefixing "የ-"). [37] presented case indicator as shown in the Table 4.

Subjective case	Ending of noun	Person	Number	Gender	Possessive case
በግ	Ending with Consonant	First	Singular		በግ-ኤ(በጌ)
			Plural		በግ-አችን(በጋችን)
		Second	Singular	Masculine	በግ-ህ(በግህ)
				Feminine	በግ-ሽ(በግች)
			Plural		በግ-አችሁ(በጋችሁ)
		Third	Singular	Masculine	በግ-ኡ(በጉ)
				Feminine	በግ-ዋ(በግዋ)
			Plural		በግ-አችን(በጋችው)
አህያ	Ending with	First	Singular		አህያ-ዬ(አህያዬ)

Vowel	plural		አህያ-አችን(አህያችን)
Second	Singular	Masculine	አህያ-ህ(አህያህ)
		Feminine	አህያ-ሽ(አህያሽ)
	Plural		አህያ-አችሁ(አህያችሁ)
Third	Singular	Masculine	አህያ-ው·(አህያው·)
		Feminine	አህያ-ዋ(አህያዋ)
	Plural		አህያ-አቸው·(አህያቸው·)

Table 2 Inflectional behavior of Amharic verbs (case indicator)

Adjectives

Adjectives are marked by numbers, gender, definiteness, and case[35]. The number marks are "-አች" for consonant ending and "-ዎች' for vowel ending. Repetition of consonant could also mark the plural form of adjectives like "ረዥም" to "ረዥ-ኣ- ዥም"[ረጓዝም]. "-ኢት" is used for gender marker and "-ን" is used to mark case (objective case). The definiteness marker is marked by using morphemes or vowels based on the number, gender or ending of the adjective. The representations of definiteness marker are discussed below[37].

Indefinite Adjectives	Ending of Adjectives	Number	Gender	Definite Adjectives
አዲስ	Consonant	Singular	Feminine	አዲስ-ዋ(አዲስዋ) / አዲስ-ኢቱ(አዲሲቱ)
			Masculine	አዲስ-ኡ(አዲሱ)
		Plural		አዳስ-አች-ኡ(አዲሶቹ)

አሮጌ	Vowel	Singular	Feminine	አሮጌ-ው(አሮጌው)/ አሮጌ-ይቱ(አሮጌይቱ)
	[redacted]		Masculine [redacted]	
		Plural		አሮጌ-ኦች-ኡ(አሮጌዎቹ)

Table 3 Inflectional behavior of Amharic verbs (definiteness marker)

2.3.1.2. Derivational behavior

Noun

Nouns are derived from other nouns, adjectives, roots, stems, and the infinitive form of a verb by affixation and intercalation[16]. The "-ነት", "-ኧኛ", "-ኧት", "-አዊ", "-ተኛ" "-ኛ" and" ባለ-"affixes are used to derive nouns from other nouns. A noun that is derived from adjective will take "-ኧት" and "-ነት" suffixes. Nouns can also be derived from verbal roots by intercalation and affixation. For instance "ነገር" is derived from "ን-ግ-ር" by "ን-ኧ-ግ-ኧ-ር" pattern of derivation.

Adjectives

Adjectives are derived from nouns, stems or verbal roots [16]. The suffixes "-ኣም" "-ኧኛ" "-አዊ" "-ኣማ" are used in the derivation of adjectives from nouns. Adjectives also derived by attaching morphemes to bound stem using "-ኣ" "-ኡ" & "-ኢታ" suffixes. Adjective those are derived from verbal roots by intercalation and affixation like "ደረቅ" is derived from "ድ-ር-ቅ" by "ድ-ኧ-ር-ኧ-ቅ" pattern of derivation.

Verb

Verbs can be derived from verbal root by affixing the vowel "ኧ" to produce CVCCVC, for instance, "ስብር" to "ስኧ-ብብኧር" [ስበር] and by repeating penultimate consonants and affixing the vowels" ኧ" and "ኣ" to produce CVCVCCVC-, e.g "ፍልግ" to "ፈላለግ" by "ፍ-ኧልኣልልኧግ" derivation pattern. Verbs can also be derived from the verbal stem by affixing morphemes like "ተ" "አስ" or "አ".

also agree on number and gender. The other agreement is an adverb – verb agreement. Usually, adverbs indicate time, therefore adverb and verb need to agree on tense. This is discussed in detail below.

2.3.2.1. Subject-verb agreement

The subject is the reigning parts of every sentence and in every sentence; subject precedes the attribute and the verb. Verbs describe the action by adding a remark on mood, tense and gender [38]. Subject and verb need to be agreed on gender, number, and person.

Subject of a sentence is presented by 'S', verb of a sentence is presented by 'V' and 'O' for object of a sentence. '3P', '2P' and '1P' stands for third , second and first person feature marker. Number feature markers are represented by 'Sing' for singular and 'PL' for plural whereas; 'M' and 'F' stands for masculine and feminine gender marker.

For example. ኦሷ መፅሐፎቹንሸጠች።(She sold the books)

S-3PSingF O-PL V-S3PSingF.

Here the subject is "She" which is a third person singular feminine and the object "books" which is a noun with plural marker "-och". Finally a verb "sold" has a feature which indicates a third person singular feminine marker. We can see that the subject and the verb are the same in number feature marker which is singular, in person agreement both indicates the third person, and also both have feminine in gender agreement.

Let's take the following example:

ኦሷ መፅሐፎቹንሸጠ።(She sold the books)

S-3PSingF O-PL V-S3PSingM.

This sentence is similar to the previous one but it is written incorrectly because the subject and the verb do not agree in gender marker. The subject indicates a third person singular feminine marker whereas; the verb indicates a third person singular masculine maker. This is grammatically invalid Amharic sentence.

Consider the verb "መጥተዋል"This verb has a third person plural marker or it can be a third person singular polite marker. Form this, we can understand Amharic verbs indicate a polite linguistic feature marker. Therefore, the verb "መጥተዋል" have a third person plural marker if the subject is in plural otherwise it can take a third person singular polite marker if the subject indicates a polite marker. For example, እሳቸውወደቤትመጥተዋል። */sacew wede bEt meTtewal /*

S-3PSingPolite Prep N V[S3PSingPolite]

2.3.2.2. Object – verb agreement

Object and verb take the same condition as subject-verb agreement. It needs to be agreed on gender, number, and person.

For example. ልጅቷ ብርጭቆዎቹንሰበረቻቸው።።

The child glasses breaks (the child(she) breaks the glasses)

S-3PSingF O-PL V[S3PSingF O3PPL]

The above example explains the object and verb agreement in number. The object "glasses" is in plural form and the verb "breaks" has a plural object marker.

Example: እሷ ለልጁ መፅሀፎቹንሰጠችው።።(she gave books to the child (he).)

S-3PSingF D_O-3PSingM ID_O-NPL V[S3PSingF O3PSingM]

The above example demonstrates the gender feature agreement of object and verb. The verb 'ሰጠችው'indicate a subject and object marker. The verb has features that indicate the subject agreement which is third person singular feminine marker and third person singular masculine object marker. The direct object ለልጁ is a third person singular masculine which agrees with the verb-object marker. Therefore the sentence is grammatically correct.

2.3.2.3. Adverb - verb agreement

Adverbs are modifiers of verbs. Adverbs can be classified into time, place and circumstance marker. Time indicates a given action in which it takes place. Verbs also indicate the time at

"ዛሬ" (today) which is in the future tense and the verb "ይመጣል" is also in a future tense. Therefore adverb and the verb agree on time but if we say "ወደቤቱትላንትይመጣል።።" (he will come yesterday). This is grammatically incorrect because the adverb "ትላንት" and "ይመጣል" do not agree in time. The adverb indicates a past tense and the verb indicates a future activity.

2.3.2.4. Adjective – subject agreement

Adjectives modify nouns by appearing before them. Adjective often agrees with the subject in gender, number, and case. In number agreement, the noun that the adjective modifies can be in plural form but the adjective can either be in singular or plural form. For instance, ደህናመፅሐፍትmeaning good books, ደህናis an adjective singular form whereas መፅሐፍትis a noun plural form. Plural form of adjective can be formed by adding 'och' in agreement with its noun but it is only essential when it also has to carry definite article 'አዳዲሶቹእስራሶች', "new pencils" or by reduplicating one of their letters. 'big[ትልቅ] can be written as [ትላልቅ][16]

Adjectives most frequently used in the masculine form. The masculine form of adjectives can modify a feminine noun but not vice versa. For example ክፉሴት(bad girl), in this phrase "ክፉ" is an adjective with masculine marker and the noun "ሴት" is in singular feminine form. But we cannot say ንፅህትወንድ(clean boy) in which "ንፅህትis an adjective in singular feminine form and the noun "ወንድ"is masculine singular form [38] .

2.3.2.5. Word order

The usual word order of Amharic is subject-object-verb (SOV). However, if the object is tropicalized it may precede the subject; as a result, it will have object-subject-verb (OSV) order. The same sentence may be equally expressed by OSV. For example 'ጌታቸውቤቱንገዛ' *Giitacaw betun gazza* 'Getachew(S) house(O, def.) buy (V , past)' and 'ቤቱንጌታቸው ገዛው' *Betun Giitacaw gazzaw* "OSV" equally mean 'Getachew bought the house'.

Example: (a) ወርቁ ዮሃንስ ከፍሉን እንደሚረብሸ ያምናለል(*Worku belives Yohannes will disturb the class.)*

O[NP NP V] S-NP V

(a) is in an SOV word order. Such a pattern can be obscure the relationship, especially in a more complex sentence with several modifiers. It is perhaps this possibility which motivates the OSV pattern of a sentence like in (b).

approaches. The reviewed papers are selected based on the approach/methods and the language family they fall in. This chapter is classified in two sub topic. The first topic discussed the review of grammar checker for foreign languages and the second topic discussed a review of grammar checker for the local languages.

3.1. Grammar checker for the foreign languages

3.1.1. A rule-based style and grammar checker for the English language

A purely rule-based open source grammar and style checker for English was discussed by [8]. The style and grammar checker described in the paper takes a text and returns a list of possible errors. QTAG (a freely available probabilistic part-of-speech tagger for non-commercial use) was used for part-of-speech tagging along with a rule-based module to help the tagger by eliminating some of the ambiguous tags before sending it to the tagger. Many words can have different POS tags depending on their context. After POS tagging, each sentence is split into chunks, a rule-based phrase chunking, i.e. a set of rules were defined that described which POS tag sequences would constitute a phrase. It then applied manually developed grammar checking rules on the POS tagged and phrase chunked-text. The grammar checker has 54 pre-defined grammar rules, which are simply a sequence of tokens to be matched. If such a pattern was found in the input text, the input is termed as erroneous. An error message is going to be displayed explaining what was wrong in the input, suggestions (if possible) to correct the error and example sentences displaying an incorrect and a correct sentence, for the particular error. There were 54 grammar rules, 81 false friend pairs, 5 style rules, and 4 built-in Python rules in this style and grammar checker.

To evaluate the system two corpora were prepared; a corpus that only contains sentences with errors and a corpus that contains very few errors (BNC's texts). When the checker is evaluated with this text, it claimed 16 errors in 75,900 sentences. However, most of these errors are false alarms. The checker also evaluated using another corpus which contains sentences with errors and compared with Microsoft Word 2000 checker. The grammar checker detected 42 errors whereas MS-word detects 49 errors. But, this style and grammar checker identifies errors in the

3.1.2. Arabic Gram Check: a grammar checker for Arabic

Grammar checker has embedded in a various word processing softwares and because of the growth of word processor technology lead to the whole class of writing errors. Errors are categorized in to two when using a word processor: those are mechanic errors and cognitive errors. Mechanic editing errors are due to cut-and-paste or Insertion–deletion operation.

- Partially deleting old text when inserting new text
- Misspellings that accidentally produce real words
- Selecting the wrong replacement text with a spell checker and
- Excessive letter or missed letters are the types of mechanical errors.

Cognitive errors occur due to lack of competence on the part of the language users to write a sentence that complies with the grammar rules, which can be corrected by replacing an existing word, inserting a new word, or moving one or more words.

Cognitive errors occur due to lack Arabic grammar is a very complex subject of study; even Arabic-speaking people nowadays are not fully familiar with the grammar of their own language. This difficulty comes from (1) the length of the sentence and the complex Arabic syntax; (2) the omission of diacritics (vowels) in written Arabic (3) the free-word order nature of Arabic sentence; and (4) the presence of an elliptic personal pronoun [21].

The main features of Arabic GramCheck are that it (1) performs a complete grammatical analysis of sentences, and (2) checks the sentence for common grammatical errors, describes the problem, and offers suggestions for improvement. The grammar checker is basically composed of two parts: an Arabic morphological analyzer and a syntactic parser extended to include a grammatical checking handler.

context-sensitive knowledge about the relation between a stem and inflectional additions. An exhaustive search to traverse the ATN generates all the possible interpretations of an inflected Arabic word. The morphological analyzer is implemented in Prolog and integrated with the parser. The morphological analyzer consists of three modules: analyzer module, a lexical disambiguation module, and a feature extraction module.

Arabic grammatical checker can be characterized as unification-based grammar (UBG) formalism. UBG grammar checker can be described as a violation of formal constraints (i.e. intra-phrasal and inter-phrasal). The central formal operation in UBG formalism is a unification of feature structures. During the construction of the Arabic parser, feature structures are translated into Prolog terms. Because of this translation step, parsing can make use of Prolog's built-in term unification.

For the evaluation 100 Arabic sentences are used to test the system. The set included both grammatical and ungrammatical sentences, taking into consideration the coverage of both the grammar rules and the grammatical errors handled by Arabic GramCheck. Of the 100 Arabic sentences, there were 10 grammatically correct sentences and 90 incorrect sentences. The average sentence length was four words and the longest sentence was 24 words long. The parser was capable of successfully parsing the longest sentence. The system includes 162 grammar rules.

Arabic gramcheck was compared to commercially available grammar checker. 96% were correctly detected by Arabic gramcheck whereas, 39% was detected by the commercially available grammar check. And 3% of the grammatical checking of Arabic GramCheck was wrong compared with 43% of the commercially available Arabic grammar checker.

It can be concluded that Arabic GramCheck was shown to be superior to the commercially available Arabic grammar checker. The reason for this is that Arabic GramCheck is more accurate at detecting cognitive errors.

uses surface grammar rules to check grammatical constructions in Swedish. The system combines probabilistic and rule-based methods to achieve high efficiency and robustness [40].

Errors that are handled by Granska system form various error types are frequent errors and common grammatical errors but also on the descriptions of correct grammatical constructions, and on decisions of which errors that are plausible to detect with the technology chosen. The system mainly focus on split compound error, internal NP disagreement and disagreement errors between the subject and the predicate which involves long-distance dependencies

Granska system consists of the tokenizer, POS tagging, error rule matching and suggestion of grammatically correct sentence. In POS tagging module second order hidden Markov model is used. Error rule matching contains a help rule and grammatically incorrect (error) rule is stated. Help rules are rules that help POS tagging module if there is incorrectly POS tagged. This helps to decrease the error rate that is caused by incorrect POS tag. In error rules, errors that are not allowed in Swedish agreement are stated. If a certain condition is matched then it is considered as an invalid sentence. For the suggestion module rules are stated if a certain condition matches the error rule matching then the possible error correction is applied. But grammatically invalid sentences can have more than one error correction; therefore, the ranking is implemented to take the likable error correction alternatives.

POS tag was evaluated and observed that unseen texts, 97% of the words are correctly tagged and unknown words are correctly tagged in 93% of the cases. 250 error rules are generated for 20 rule categories. The system was compared with other systems and achieved a high performance. High-performance matters in practical applications, where the grammar checker, for instance, should be run in an interactive mode, checking every new sentence written. Memory usage is also important in practical applications, and Granska consumes about 22 MB RAM.

of Speech (POS) Tagger Module, Chunker and Parser Module, Grammatical Relation Finder, Syntax checker Module and suggestion creating module components [33]

After the tokenizer module which splits the input file into paragraphs, sentence, and words. Morphological Analyzer Module will be carried out which breaks the input words as root or breaking down the input words into its constituent morphs, root word, and associated affixes. The word breaking is done on the basis of word breaking rules available for Nepali. In addition to this, this module also performs the first level Parts of Speech (POS) tagging of the morphs as per the information gathered from the free morpheme and affix lists. The term first level POS tagging implies that further processing would need to be carried out for hundred percent correct POS tagging.

Parts of Speech (POS) Tagger Module; The module "POS Tagger Module" POS tags the untagged and POS tag undetermined tokenized words. Assigning a particular POS category to a word out of the several possible thus Chunker and Parser Module identifies the chunks or phrases from the POS tagged words in the sentence of the input text file. For this, apart from the two files, the free morpheme list, and the affix list, we would need a set of context-free grammar rules or productions.

Grammatical Relation Finder; the assignment of grammatical roles is based on the agreement phenomenon in the Nepali language. The Nepali language has the following agreement patterns: SUBJECT-VERB agreement; Agreement between Modifier and Head in the Noun Phrase. Syntax checker Module; the module "Syntax checker module" checks whether the SUBJECT OBJECT VERB (SOV) pattern and responsible for checking the agreement between the words. Suggestion creating module consults the Chunker and Parser Module, Grammatical Relation Finder Module and the Syntax Checker Module and suggests a different sentence structure

3.2.1. A rule-based approach for Afan-Oromo grammar checker

To develop a grammar checker for Afan-Oromo language, a rule-based approach was chosen. Tokenization, POS tagging, stemming, grammatical relation finder and suggestion creating module are the major steps to build grammar checker for Afan-Oromo language [15].

Tokenization is used to split paragraphs into sentence and sentence into words which will be used as an input for the next stage i.e. POS tagging. Each word in the corpus is tagged with its appropriate part of speech using hidden Markov model (HMM). The next stage is the stemmer module. This algorithm is a procedure that reduces all words with the same stem to a common form by stripping of its derivational and inflectional suffixes. The Afan-Oromo stammer is based on a series of steps that each removes a certain type of affix byway of substitution rules. These rules only apply when certain conditions hold. In grammatical relation finder step, 123 rules were constructed and presented to identify subject and verb, subject and adjective, main verb and subordinate verb in terms of number, gender, and tense agreement. The last stage is suggestion creating module which gives alternatives of in two directions. If a subject-verb disagreement occurs in number, for instance, the suggestion stage will give an alternative to correct subject in number leaving the verb as it is or the other way around.

This research work gave attention mainly on subject-verb agreement, subject – adjective agreement and main verb and subordinate verb agreement. The system showed 88.89% of precision and 80% of recall.

3.2.2. Grammar checker for the Amharic language

Amharic grammar checker was developed by Aynadis Temesgen in 2013. To check the grammatical correctness of Amharic sentence, sentences are classified in to simple and complex sentence. The system was implemented by using two different approaches, Rule based and statistical based approaches.

Rule-based approach is conducted for only simple sentence. Rule based grammatical checker accepts input text and identifies grammatically correct and incorrect sentence. Rule-based

tokenized words will be sent to a morphological analyzer which is the second stage. HornMorph is Amharic morphological analyzer which is used to identify the linguistic meaning of each word. The next stage is grammatical relationship finder which identifies the relation between subject-verb and object-verb in terms of person, number, and gender. This relation is identified based on the output of morphological analyzer. The language model is the fourth step in building Amharic grammar checker which is storage of that holds grammatical rules. Rules describe **what must not occur** in Amharic sentences or **incorrect grammatical sentence structure in the form of words pattern**. This means, if a sentence matches with any of rules then it will be considered as a grammatically incorrect sentence. Grammar rule checker module matches the patterns extracted in the grammatical relation finder module against the rules in the language model. The grammatical errors in the sentence will be checked based on the rules in the language model.

To evaluate rule-based Amharic grammar checker, 50 simple sentences are self-prepared which include grammatically correct and incorrect sentence. Those sentences also include multiple errors per sentence. The checker gives 92.45% of Precision and recall: 94.23%.

A second approach is a **statistical approach** which is implemented for both simple and complex sentence. **Statistical approach** has two major tasks. The first one is training the system by extracting patterns with different 'N' values (bi-gram and tri-gram) and the second one is checking grammatical error using the extracted pattern set on subject-verb, object-verb and adverb - verb agreement is discussed. Manually POS tagged corpus is prepared for both training and test set. The probability will be calculated for POS tag sequence which will be extracted based on bi-gram and tri-gram values and grammatical agreement is checked with the possible sequence of POS tag in terms of person, gender and number will be extracted.

The system is evaluated in both simple and complex sentence. 50 simple sentences were prepared (the same test set which is used for the rule-based approach). Forical error per sentence. The evaluation of the system on rule-based approach gives Precision: 92.45 % Recall: 94.23% and in Statistical ach gives Precision: 67.14% and Recall: 90.38%. The system detects multiple errors at once but gives false alarm due to incomplete rules and quality of statistical data.

sentence, the performance of the statistical approach was promising. Therefore by taking the good qualities of the two approaches, this research investigated the use of a probabilistic method with rule-based approach for Amharic grammar checker.

is implemented to address identified problem using problem identification. Design science has six key process or stages. Those are **problem identification and motivation, objectives of the solution, design and development, demonstration, evaluation and communication.**

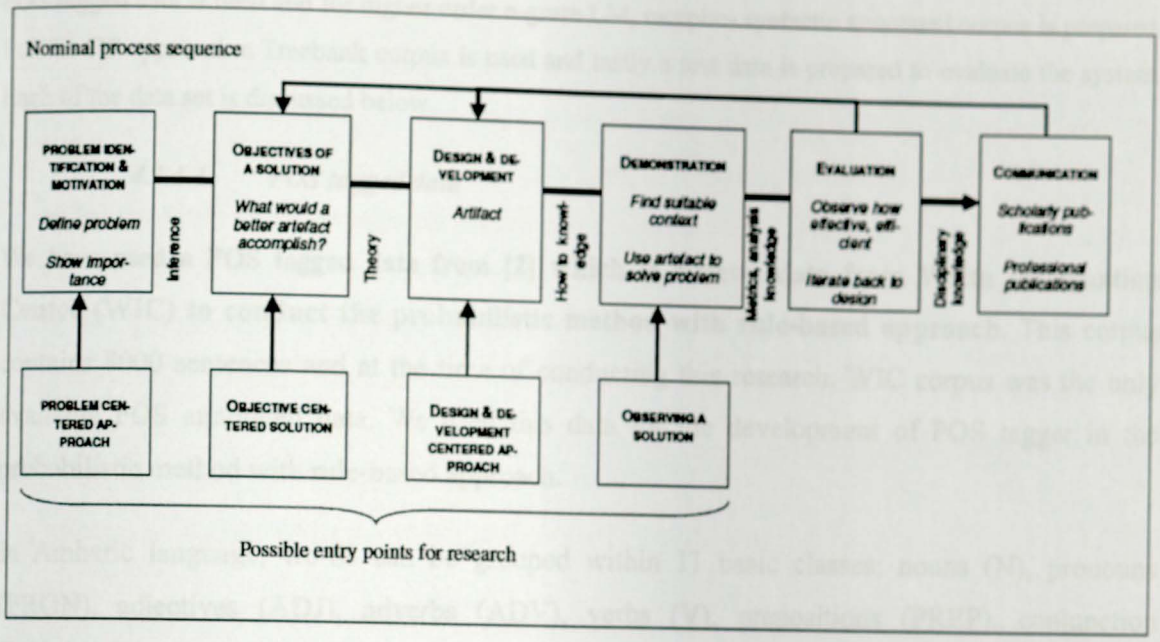


Figure 1 architecture of design science

4.1. Problem identification and motivation

It is the first step and a key process in design science. It is a process of identifying the key area and needs consideration while doing a research on the topic. To identify and understand the problem we are working on, we did literature review on the topic to have the fundamental concepts and understand the approaches, algorithms, and methods that are used to develop a grammar checker.

4.2. Objectives of the solution

After having the problems identified, the objectives of this study are defined well and keeping these objectives in mind, we continue to the design and development of the study.

4.3.1. Data collection and preparation

To conduct the three grammar checker approach data were prepared and used from different sources. For each of the approach, different data set were used. For the probabilistic method with rule-based approach, POS tagged data is used and for higher order n-gram LM, morpho- syntactic annotated corpus is prepared. For the DP approach a Treebank corpus is used and lastly a test data is prepared to evaluate the system. Each of the data set is discussed below.

4.3.1.1. POS tagged data

We have used a POS tagged data from [2] which is a news data from Walta Information Center (WIC) to conduct the probabilistic method with rule-based approach. This corpus contains 8000 sentences and at the time of conducting this research, WIC corpus was the only available POS annotated data. We used this data for the development of POS tagger in the probabilistic method with rule-based approach.

In Amharic language, words can be grouped within 11 basic classes: nouns (N), pronouns (PRON), adjectives (ADJ), adverbs (ADV), verbs (V), prepositions (PREP), conjunction (CONJ), interjection (INT), punctuation (PUNC), numeral (NUM) and UNC which stands for unclassified and used for words which are difficult to place in any of the classes [35]. Some of these basic classes are further subdivided into a total of 31 POS tags presented in appendix A.

> "The number of tag-sets, which is proposed based on the orthographic word, is 31. Compound tag-sets were used for those words with prepositions and conjunctions. Even though some subclasses of a given category (like noun and verb) are distinguished, the distinction no longer exists when these forms attach prepositions and/or conjunctions. However, they propose independent tags for both prepositions and conjunctions without favoring segmentation of these clitics [2]."

4.3.1.2. Chunked data

We have prepared a manually chunked data for chunking phase by taking 509 sentences from WIC corpus and manually chunked using IOB2 format. IOB2 format is short for inside-

and I-prefix before a tag indicates the tag is inside of the chunk. B-tag is used at the beginning of every chunk i.e. all chunks start with the B-tag. O-tag indicates that a token belongs to no chunk[41]. This data is also prepared because the unavailability of chunked data.

4.3.1.3. Morpho- syntactic annotated corpus

To build higher order n-gram LM, WIC news corpus is further annotated by their appropriate grammatical feature markers i.e. number, person and gender marker by using HornMorpho analyzer. HornMorpho is used to give a **morpho-syntactic** feature to each word [42]. Morpho-syntactic feature is a feature which is relevant to syntax. Gender, person and number are involved in the language. For a feature to be 'relevant to syntax' means that it involves in either syntactic agreement or government of the language. **We have prepared this data because POS tag annotated with syntax feature marker was not available even though the previous researcher prepared data by using similar format and this corpus is used to implement higher order n-gram approach only.** The corpus is presented in specific template. This template is developed based the affix order mentioned in section 2.3.1.4.

The first template is used for words which does not have object marker indicator while the second template is used for word which have both subject and object marker.

<POS | number_marker_subject| person_marker_subject| gender_marker_subject >

Or

<POS | number_marker_subject ^ number_marker_for_object | person_marker_subject ^ person_marker_object | gender_marker_subject ^ gender_marker_object >

Each of the grammatical features are separated by '|" sign and to differentiate the subject and object grammatical feature marker, '^' is used.

Example በአዳማ<NP> ልዩ<ADJ> በተጠናቀቀው<VP|sing^sing|3^3|masc^masc> የበጀት<NP> በ6.8 ሚሊየን<NUMP> ከተጀመሩት<VP|plur|3|NN> 12<NUMCR> ፕሮጀክቶች<N|plur|NN|NN> መካከል<PREP> አብዛኞቹ<ADJ|plur|3|masc> ተጠናቅቀው<V|plur|3|masc> 50ሺ<NUMCR>

The word በአዳሜ has <*NP*>tag, which indicates the POS tag ***Noun Phrase*** /'*NP*' and '*ልዩ*' *has adjective /'ADJ' POS tag.* The third word in the sentence '*በተጣናቀቀው*' has <*VP|sing^sing|3^3|masc^masc*>tag, the first slot indicates the POS tag ***Verb Phrase*** /'*VP*' and the second slot indicates ***singular***/'*sing*' subject number marker and ***singular***/'*sing*' object number marker, the third slot indicates the '*3rd*' person subject marker and '*3rd*' person object marker, the last slot indicates **masc**uline gender marker *'masc'* and **masc**uline object marker '*masc*' and so forth.

4.3.1.4. Treebank corpus

Treebank corpus was used from [3] to train and test dependency parser. A Treebank can be viewed as linguistically annotated corpus that includes grammatical analysis beyond part of speech level [3]. The corpus includes 1000 sentences which are annotated for POS tag feature and morphological feature. Each of the content words is manually segmented, POS tagged and analyzed for morphological features. The corpus is prepared by separating the functional word or clitics form phonological host. Prepositions, the Possessive marker or pronominal genitive markers, Definite marker, accusative marker, conjunction, Negation, Auxiliaries, Relative pronouns, Nominal clause marker, Subject and object pronominal agreement markers were separated from the content word and are considered as clitics [3]. For example ከየቤቱና/kəjjəbetunna/ "from each house and", includes the preposition ከ- /kə-/ "from", reduced form of the distributive marker አየ- /ijjə/ "each", ቤት/bet/ "house", the definite marker -ኡ/-u / "the" and the conjunction -ና/-nna/ "and". The clitics have syntactic roles that indicate grammatical relations with the content words. In order to show the syntactic relation between clitics and content words56 POS tag sets were compiled.

The Treebank data is presented in CoNLL-X format where all the sentences are in one text file and they are separated by a blank line after each sentence. A sentence consists of one or more tokens. Each token is represented on one line, consisting of 10 fields.Fields are separated from each other by a TAB. The fields are ID, FORM, LEMMA, CPOSTAG, POSTAG, FEATS, HEAD, DEPREL, PHEAD, and PDEPREL (the fields are explained in appendix C). Out of the

available, an underscore is present. Except FEATS column, all other columns are fixed. But, in FEATS column we can have any useful information other than the information encoded in the fixed columns.

Consider the first row in the CoNLL format presented below in Table 6. '1' is the ID of the node in the sentence.' መጽሐፍ' is the word and a LEMMA form, 'NOUN, and 'NOUN' are the CPOSTAG and POSTAG respectively. HEAD and DEPREL are '4' and 'obj' (stands for object) respectively. All these are fixed columns. FEATS column is used to represent the extra information in the form of gender, number, person, and case (morpho-syntactic features).

1	መጽሐፍ	መጽሐፍ	NOUN	NOUN	_	4	obj	_	Translit=mäcəħäfə|LTranslit=mäcəħäfə
2	ኡ	ኡ	DET	DET	_	1	det	_	Translit='u|LTranslit='u
3	ን	ን	PART	ACC	_	1	case	_	Translit=nə|LTranslit=nə
4	አስያዝ	አስያዝ	VERB	VERB	Voice=Cau	0	root	_	Translit='äsəjazə|LTranslit='äsəjazə
5	እኔ	እኔ	PRON	SUBJC	Gender=Masc|Number=Sing|Person=3	4	nsubj	_	Translit='ă|LTranslit='ă
6	አት	አት	PRON	OBJC	Gender=Fem|Number=Sing|Person=3	4	iobj	_	Translit='atə|LTranslit='atə
7	።	።	PUNCT	PUNCT	_	4	punct	_	Translit=.|LTranslit=.

Table 4 example on DP data format

We have prepared test data which contains 150 sentences form Amharic text books which contains grammatically correct and incorrect sentences. The incorrect sentence is prepared by the help of linguist and this test data is used to evaluate the grammar checker system.

4.3.2. Preprocessing

After the data collection, preprocessing is the next module that follows in order to make the corpus that we have on hand is in the appropriate format.

Spell checker: To guaranty the corpus quality, Abyssinica Amharic spell checker is used to check the spelling in the WIC corpus.

Morpho- syntactic annotated corpus; each of the words is analyzed by HornMorpho analyzer to add grammatical tag on the corpus i.e. number, gender and person feature marker tags. The output of the analyzer is presented by using the above mentioned template. This corpus is only prepared for conducting higher order n-gram LM.

MaltOptimizer-data validation: The Treebank corpus was cleaned by removing comments and by using MaltOptimizer tool to check whether the corpus is in valid CoNLL-X format.

4.4. Demonstration

As reviewed in chapter two and three of this thesis, there are different types of Grammar checking approaches and algorithms. In this section, we discuss the different techniques we used to deploy our system.

4.4.1. Approach

We construct an automatic grammar checking system by using probabilistic method with rule-based approach. In the statistical (probabilistic) method, POS tagging and chunker is developed. We have conducted POS tagging and chunking using HMM and TnT algorithms and better performance score for each of the module is taken as a training algorithm. Rules are crafted to check the grammatical correctness of a given sentence. The set of rules hold grammatically

Tri-, quadra-, and penta-gram LM is developed by using good Turing (the default) smoothing technique. Niver standard pseudo-projective is selected as best fit parsing algorithm by MaltOptimizer toolkit and rules were also used in the development of DP to check the grammatical correctness of the input sentence.

4.4.2. Tools

To develop grammar checker for Amharic language appropriate tools were selected. The tools used in this research will be discus as follows.

- For the preprocessing stage, i.e. transforming the raw date in to clean data set before feeding to the learning algorithm; Abyssinica open source spell check software is used to check the spelling correctness of Amharic sentences.
- Amharic morphological analyzer, HornMorpho, is used to give a **morpho-syntactic** feature to each word in our corpus [42]. HornMorpho is currently available morphological analyzer for Amharic, Tigrigna, and Afan-Oromo languages [42].
- To build n-gram LM we have used SRILM toolkit. SRILM is freely available toolkit which supports creation and evaluation of a variety of language model types based on N-gram statistics, as well as several related tasks, such as statistical tagging and manipulation of N-best lists and word lattices [43].
- DP is built using MaltParser which is a data-driven parser generator for dependency parsing. It is freely available for research and educational purposes and has been evaluated empirically on Swedish, English, Czech, Danish and Bulgarian language [44]. MaltOptimizer and MaltEval are used to for data validation and evaluation of the system respectively. MaltOptimizer is an interactive system that first performs an analysis of the training set in order to select a suitable starting point for optimization and then guides the user through the optimization of parsing algorithm, feature model, and learning algorithm.

complete tasks and the familiarity of the researcher with this programming language.

4.5. Evaluation

To evaluate the system precision, recall, and accuracy is used. Precision refers to the percentage of flagged grammatical errors that are in fact grammatical errors. Recall refers to the percentage of grammatical errors occurred in the text and those are in fact flagged. Accuracy deals with how the system precisely detects grammatically correct and incorrect sentences. This evaluation metric is chosen because the researcher is familiar with it and can be computed easily.

$$Recall = \frac{numberoferroscorrectlydetected}{numberoferrors}$$

$$Precision \frac{numberoferroscorrectlydetected}{numberoferrosdetected}$$

$$Accuracy = \frac{numberoferrorscorrectlydetected}{total}$$

Using MaltEval toolkit, DP is evaluated by selected measurement metrics which are LAS, UAS and LA. These metrics values are used to know how many decisions did not get right. In DP approach label and heading values are important to analyze the grammatical agreement between the words (detail is presented in section 5.3) therefore the hit or miss of the labels are crucial in evaluating DP.

$$LAS = \frac{numberofcorrectheading\&lables}{numberofheading\&lables}$$

$$UAS = \frac{numberofcorrectheading}{numberofheading}$$

$$LA = \frac{numberofcorrectlables}{numberoflables}$$

4.6. Communication

Communication is being done through this thesis. In addition, the researcher is planning to publish in journal/article.

The design and implementation of the three approaches, the probabilistic method with rule-based approach, higher order n-gram, and dependency parser are presented in detail.

5.1. Probabilistic method with rule-based approach for Amharic Grammar checker

The probabilistic method with rule-based approach contains two major phases. The first phase uses a statistical (probabilistic) method for POS tagging and shallow parsing (chunking). Shallow parsing is used for identification of correct SOV word order in the Amharic language. The second phase is the rule-based approach which accepts the output of the first phase as input and analyzes it by using crafted rules. The rules are manually crafted rules that contain a correct grammar of the Amharic language.

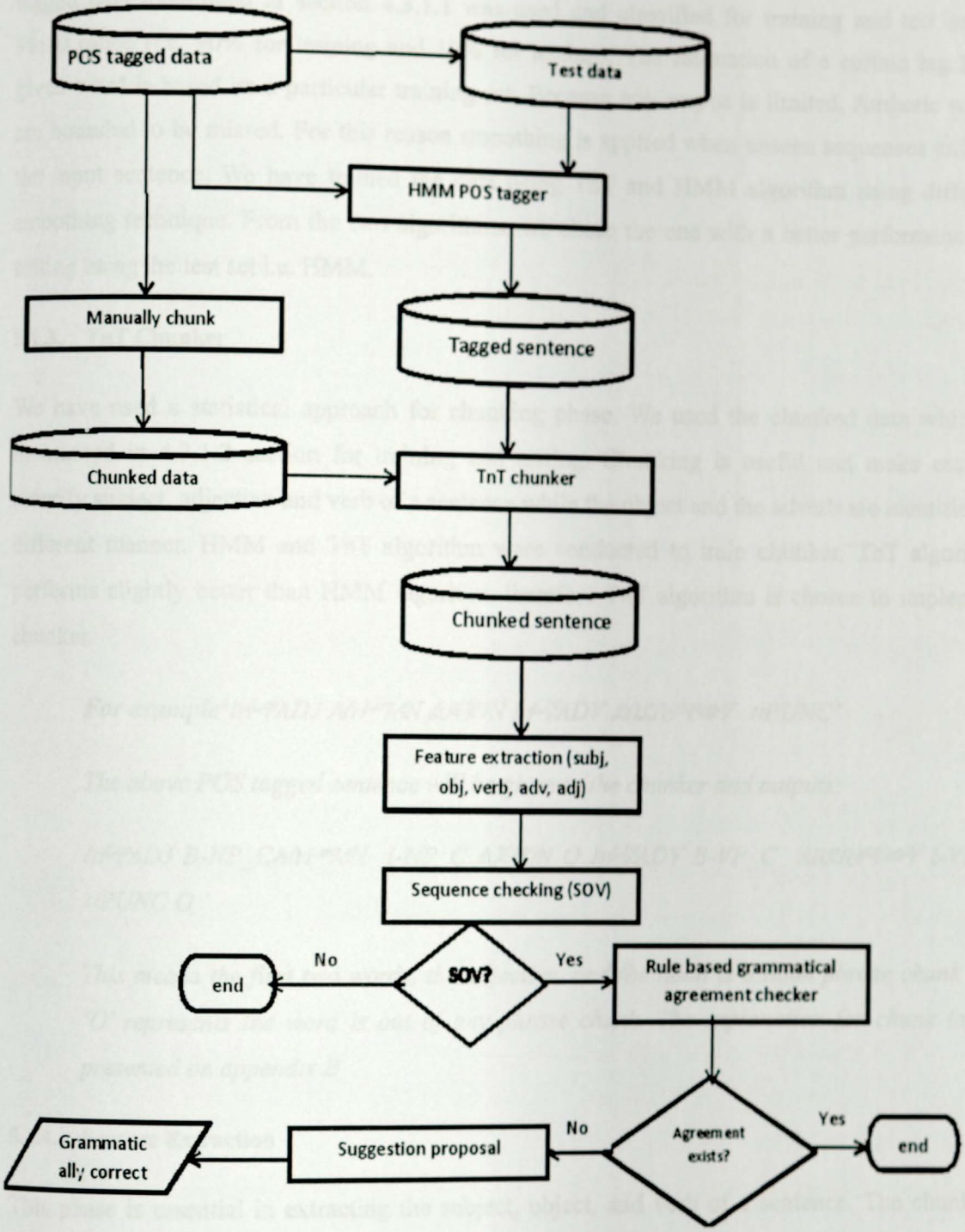


Figure 2 Architecture of probabilistic method with rule-based approach approach

To automate the grammar checker system POS tagging is implemented. In this phase, the POS tagged data mentioned in section 4.3.1.1 was used and classified for training and test set by 90:10 ratios (i.e. 90% for training and 10% for testing). The estimation of a certain tag for a given word is based on a particular training set. Because any corpus is limited, Amharic words are bounded to be missed. For this reason smoothing is applied when unseen sequences exist in the input sentence. We have trained the data using TnT and HMM algorithm using different smoothing technique. From the two algorithms, we chose the one with a better performance by testing using the test set i.e. HMM.

5.1.3. TnT Chunker

We have used a statistical approach for chunking phase. We used the chunked data which is mentioned in 4.3.1.2 section for training and testing. Chunking is useful and make easy to identify subject, adjective and verb of a sentence while the object and the adverb are identified in different manner. HMM and TnT algorithm were conducted to train chunker. TnT algorithm performs slightly better than HMM algorithm therefore TnT algorithm is chosen to implement chunker.

For example *'ከፉዋADJ አስተማሪN ልጅቸንN ከፉኛADV ደበደበቻቸውV ::PUNC'*

The above POS tagged sentence will be given to the chunker and outputs:

ከፉዋADJ B-NP_Cአስተማሪ N I-NP_C ልጅቸንN O ከፉኛADV B-VP_C ደበደበቻቸውV I-VP_C ::PUNC O '

This means the first two words, the adjective, and the noun is a noun phrase chunk and 'O' represents the word is out of any phrase chunk. The explanation for chunk tag is presented on appendix B

5.1.4. Feature Extraction

This phase is essential in extracting the subject, object, and verb of a sentence. The chunking phase plays an important role in the identification of S-O-V sequence with in a sentence because

phases work as follows:

A subject of a sentence: for a word to be a subject of a sentence, it has to appear at the beginning of a sentence[16]. Therefore, we take the first **noun phrase chunk** then we will look for the last noun in the **noun phrase chunk** word sequence or a noun type POS tag which appears at the beginning of the sentence with 'O' chunk type.

An object of a sentence: for word to be an object of a sentence, it has to have an accusative marker 'ን' at the end of the word[38]. An object is identified with the help of HornMorpho analyzer. If the analyzer detects the word has accusative marker then the word will be taken as an object. If not then the sentence does not have an object.

A verb of a sentence: since a verb of a sentence always appears on the end of a sentence, the extraction phase will look at the last **verb phrase chunk** or a **verb** of the sentence[35].

Adjective of a sentence: if a word appears right before the subject of a sentence and its part of speech tag is an **adjective (ADJ)** then it is taken as **an adjective of the subject** in a given sentence[16].

Adverb of a sentence: the word with an **adverb (ADV)** part of speech tag is taken as an adverb of a sentence [16].

> ***Example***: *ከፉዋADJ B-NP_CአስተማሪN I-NP_C ልጅቱንN O ከፉኛADV B-VP_C ደበደበቻትውV I-VP_C ::PUNC O ' 'this was the output of the chunker module which is passed to the extraction module. As mentioned above the subject will be detected if the word appears at the beginning of a sentence and the word must have a noun type chunk and be in the first noun phrase chunk. In this example, 'አስተማሪ' is the subject of the sentence. The adjective is detected if it appears before the subject of the sentence in which the extraction module detects as a subject of a sentence and has an adjective POS tag. In the above sentence, 'ከፉዋ' is the adjective of the sentence. The object of a sentence is detected if the word has the object marker 'ን' at the end of the word. To be able to detect whether the 'ን' marker is object marker or not, the HornMorpho analyzer was used. If the analyzer returns 'accusative' for the given word, then the word will be taken as an object of a sentence.*

detected as a verb of the sentence. Therefore,

Adjective: ከፋዋ

Subject: አስተማሪ

Object: ልጅቿን

Verb: ደበደበቻቸው

5.1.5. SOV sequence checker

The Amharic language has a subject-object-verb word order [35]. This module checks whether the input sentence is in the correct SOV sequence. No further analysis is done if a sentence fails to have the correct word order, even though object can precede the subject results object-subject-verb order. The system only assumes the SOV word sequence as a valid sentence.

5.1.6. Rule based grammatical Agreement checker

The word that has been extracted as subject, object, verb, adjective, and adverb will be passed to this module. The **Agreement phase** will check the number, gender, person and tense feature of a given sentence. The extracted words (i.e. subject, object, adjective and verbs) are passed to the morphological analyzer, HornMorpho. The output of the analyzer will be passed to the crafted **rule** which holds a correct grammatical rule of Amharic language.

Agreement in grammatical feature should exist between subject-verb, adjective-subject and object- verb in a sentence. For instance number feature marker (plural, singular) of a subject must match with number feature marker (plural, singular) of a verb respectively. Also person feature marker (1st person, 2nd person and 3rd person) of a subject must match with person feature marker (1st person, 2nd person and 3rd person) of the verb respectively and for gender feature marker, if subject of a sentence has masculine or feminine marker then the verb must have a masculine or feminine feature marker respectively to be grammatically correct. Sentences might

subject - verb is considered as grammatically correct in gender.

// the rule which checks the agreement ***of subject and verb in gender feature.***

IF the ***subject*** *of a sentence has a* ***masculine*** *marker and* ***verb*** *of the sentence has a* ***masculine*** *marker*

The sentence is ***correct*** *in gender agreement*

ELSE IF the ***subject*** *of a sentence has a* ***feminine*** *marker and* ***verb*** *of the sentence has a* ***feminine*** *marker*

The sentence is ***correct*** *in gender agreement*

ELSE IF the ***subject*** *of a sentence has either a* ***feminine or a masculine*** *marker and the* ***verb*** *of a sentence has a* ***neutral*** *marker*

The sentence is ***correct*** *in gender agreement*

ELSE IF the ***subject*** *of a sentence has a* ***neutral*** *marker and the verb of a sentence has a* ***neutral*** *marker*

The sentence is ***correct*** *in gender agreement*

ELSE IF the ***subject*** *of a sentence has a* ***neutral*** *marker and the* ***verb*** *of a sentence has either a* ***feminine*** *or* ***a masculine*** *marker*

The sentence is ***correct*** *in gender agreement*

ELSE

The sentence is ***incorrect*** *in gender agreement*

These rules can be applied for object-verb agreement. Adjective –subject also need to be agreed on gender and number feature marker by applying the same rule mentioned above. If the sentence fails to match with the rules it will be considered as an incorrect sentence.

// the rule which checks the agreement of ***subject and verb in number feature.***

IF the ***subject*** *of a sentence has a* ***plural*** *marker and* ***verb*** *of the sentence has a* ***plural*** *marker*

The sentence is ***correct*** *in number agreement*

ELSE IF the ***subject*** *of a sentence has a* ***singular*** *marker and* ***verb*** *of the sentence has a* ***singular*** *marker*

The sentence is ***correct*** *in number agreement*

ELSE

The sentence is ***incorrect*** *in gender agreement*

*IF the **subject** of a sentence has a 1st **person** marker and **verb** of the sentence has a 1st **person** marker*

The sentence is correct in person agreement

*ELSE IF the **subject** of a sentence has a 2nd **person** marker and **verb** of the sentence has a 2nd **person** marker*

*The sentence is **correct** in person agreement*

*ELSE IF the **subject** of a sentence has **a** 3rd **person** marker and **verb** of the sentence has a 3rd **person** marker*

*The sentence is **correct** in person agreement*

ELSE

*The sentence is **incorrect** in **gender** agreement*

The subject and the adjective will first pass to HornMorpho analyzer. This is because, in the Amharic language, the adjective sometimes will indicate the feature of the noun that it modifies.

***For example**ከፉዋADJ B-NP_CኦስተማሪኤN I-NP_C ልጅቱN O ከፉኛADV B-VP_C ደበደበቻቸውV I-VP_C ::PUNC O from this sentence, the adjective, subject, object and verb of the sentence are detected and passed to the agreement phase. The adjective and the noun (subject) will be analyzed and the analyzer output is*

ከፉዋ:3, sing, fem

ኦስተማሪ:3, sing, masc the analyzer represent the person, number and gender feature maker respectively. The person and number match for both words but it differs in gender. The adjective gender marker indicates it is a feminine marker whereas the subject feature marker indicated masculine gender marker. Therefore, as we observe, the subject 'ኦስተማሪ' is feminine indicated by the adjective marker. The rule will override the gender feature marker of the subject by the adjective.

Then subject and the verb will be analyzed by HornMorpho analyzer. From the analyzer output, the number, person and gender markers of the subject and verb are identified. The agreement

as correct agreement. If it mismatches, then the words is passed to the suggestion module.

***Example**: subject 'አስተማሪ': 3, sing, fem*

Verb 'ደበደበቻቸው': subject marker 3, sing, fem

:Object marker 3, plur, neutral

The first slot indicates the person feature marker, the second slot indicates number feature marker and the last and the third slot indicates the gender feature marker. The subject now has a feminine gender marker as it is expressed in the adjective, Therefore, the subject and verb matches in person, number and gender feature marker.

The object and verb of a sentence also passes in the same procedure as the subject-verb agreement module. But here, the verb of a sentence might not hold the object feature marker (i.e. number, person, gender), when this exception occurs the object and verb is considered as correct agreement. If the verb holds the object marker, then it is checked against the agreement rule. The object and verb is checked on number, person and gender marker and if it matches it is considered as correct. The suggestion module takes place if the agreement is incorrect.

***Example**: ከፉዋADJ B-NP_CአስተማሪN I-NP_C ልጆቹንN O ከፉኗADV B-VP_C ደበደበቻቸውV I-VP_C ::PUNC O In this example, the verb '' has a subject marker of 3rd person, singular, feminine and object marker of 3rd person, plural, masculine form. The output of the analyzer is*

***Subject 'አስተማሪ':** 3, sing, fem*

***Object 'ልጆቹን':** 3, plur, neutral*

***Verb 'ደበደበቻቸው':** subject marker 3, sing, fem*

Object marker 3, plur, neutral

similar feature with the subject which is a 3rd person, singular, feminine form. The subject and verb feature match, therefore, we can conclude that the subject and the verb agreed on a person, number and gender. Then the object and verb agreement are going to be carried out. According to the analyzer output, the object has a 3rd person, plural and masculine feature marker and the verb-object marker has a 3rd person, plural, neutral form. Therefore, the object and the verb match in person, number, and gender feature marker. We can conclude that the sentence is a grammatically correct sentence.

Since HornMorpho cannot analyze adverb, we built a bi-gram probability from the collected WIC corpus. To build the bi- gram probability of the adverb, we extract all adverbs in the corpus with their immediate verbs. Then we analyze the verb and take the tense indicator from the verb and assign it to the adverb. Probability is computed and saved in the repository for each tense type i.e. perfective, imperative, jussive, and gerundive.

Therefore, when the system found an adverb on our test set, the model looks to the probability model and give the highest score to the adverb. Then the immediate verb in the test sentence will be analyzed and computed with the adverb. If the output of the analyses of the verb matches with the adverb result, then it is considered as a correct adverb – verb agreement. Incorrect if it fails to match.

For example *ልጅN O ውጤቱንN O ወደፊትADV B-VP_C አሻሽለV I-VP_C ::PUNC O*

The above sentence is the output of the chunk module which indicates the word, POS tag and the chunk tag respectively. After passing through the feature extraction and SOV sequence checker the agreement module outputs the following results.

Subject 'ልጅ':3, sing, masc

Object 'ውጤቱን':3, sing, masc

Verb 'አሻሽለ':3, sing, masc, perfective

the adverb is looked through the bi-gram probability model and tense feature of the adverb is extracted. By looking through the probability model the tense feature is extracted. The bi-gram model returns 'imperfective' for the word 'ወደደ'whereas, the verb has a 'perfective' form. Therefore, the system detects this mismatch and concludes there is adverb and verb disagreement.

Independent Pronoun

HornMorpho cannot analyze independent pronouns, therefore; we built a dictionary for independent pronoun with their respective grammatical feature. Independent pronouns are limited in number in the Amharic language this makes is easy to prepare a dictionary for this part of speech class. There is no specific tag for the independent pronoun in our corpus, therefore, the system checks if a given word is independent pronoun by checking against the dictionary. If exist, the grammatical feature is taken from the dictionary otherwise, the word passed to the HornMorpho analyzer. The following table shows the independent pronouns of Amharic language.

English	Independent pronoun	Object pronoun suffixes
I	አኔ/əne	-(ä/ə)ň
you (m. sg.)	አንተ/anta	-(ə)h
you (f. sg.)	አንቺ/anči	-(ə)š
He	አሱ/əssu	-(ä)w, -t
She	አሷ/əsswa	-at
We	እኛ/əññа	-(ä/ə)n
you (pl.)	አናንተ/ənnantä	-ačč̌əhu
They	እነሱ/ənnässu	-ačč̌äw

Table 5 independent pronoun of Amharic language

Amharic sentences can have a null subject [38]. The sentences might only have a verb or a verb with an object in the Amharic language.

Example: ራሴን፡ ያመኛል

rasen : yammaf1f1a II/ my head-acc : hurts me

'I have a headache. '

The verbs have a tense aspect marker which is third person singular, but there is no subject to which it refers. Amharic has subject-less sentences[45].

When these exception occur the system takes the subject feature marker of a verb as a subject features agreement. And the subject will be indicated with a special marker '$' to represent the null subject.

5.1.7. Suggestion

This module works with HornMorpho generator which takes the root of the Amharic verb to generate a correct verb form and stem of a noun to generate a correct noun form. The root or the stem is taken along with their agreement markers to generate the desired result. The module will only takes place when input sentence has a grammatical error that is detected by agreement module. If the above example sentence is written as follows it results a disagreement in gender.

Example: *ከፉዋADJ B-NP_C አስተማሪN I-NP_C ልጅቸንN O ከፉኗADV B-VP_C ደበደባቸውV I-VP_C ::PUNC O*

The subject of the sentence is 'አስተማሪ' and the verb of the sentence is 'ደበደባቸው' and the object of the sentence is 'ልጅቸን'. The grammatical feature of the subject is '3, sing, fem and the verb has a subject marker of '3, sing, masc' and object marker of '3, sing, neutral'. Here the person, number and gender of the subject, object and the verb are indicated respectively. In subject and verb agreement the subject marker in person is similar to the verb person feature marker and also the number marker of the subject and

marker indicates 'fem/feminine' but the verb number marker indicates 'masc/masculine'. When the disagreement occurs the suggestion module will take place.

When we look into object and verb agreement, the verb has object feature marker indicator, which is '3, sing, masc'. The object has '3, 'sing, nuteral'. Therefore, the verb and the object match in number and person since both have singular 3rd person feature marker. They also agree on gender since the verb indicates masculine and object indicate neutral form respectively.

Therefore, this sentence is grammatically incorrect because disagreement occurs between the subject and verb in gender feature marker. When the agreement rule detects the disagreement then the module will pass to suggestion phase.

The suggestion module takes the root of the verb and the agreement feature to generate the desired output.

Example: *from the above example the root of the verb 'ደበደበ' is 'dbdb'. The root of the verb and the agreement features of the subject and the object is passed to the HornMorpho generator for the desired output.*

Subject = 3+ sing+ fem, object =3+ plur + root of the verb = 'dbdb'.

The generator takes this grammatical feature and generate the desired verb form. The generator outputs 'ደበደቧቸው'.

The HornMorpho generator cannot analyze adjectives therefore; we build a dictionary for consonant and vowel characters. This dictionary helps to generate the correct word form for adjective- subject disagreement. As mentioned on 2.3.1 section subsection 2.3.1.1 adjectives inflect in number based on their consonant or vowel endings. Therefore, if the adjective – subject disagreement occurs in the sentence, the system will check whether the subject is a consonant or a vowel ending. Then based on the result, the appropriate subject form is generated.

5.2. N-gram based Amharic grammar checker

Higher order n-gram implementation is discussed in this section. The tri-, quadra- and penta-gram LM is investigated.

5.2.1. The Architecture of N-gram Language model for grammar Checking

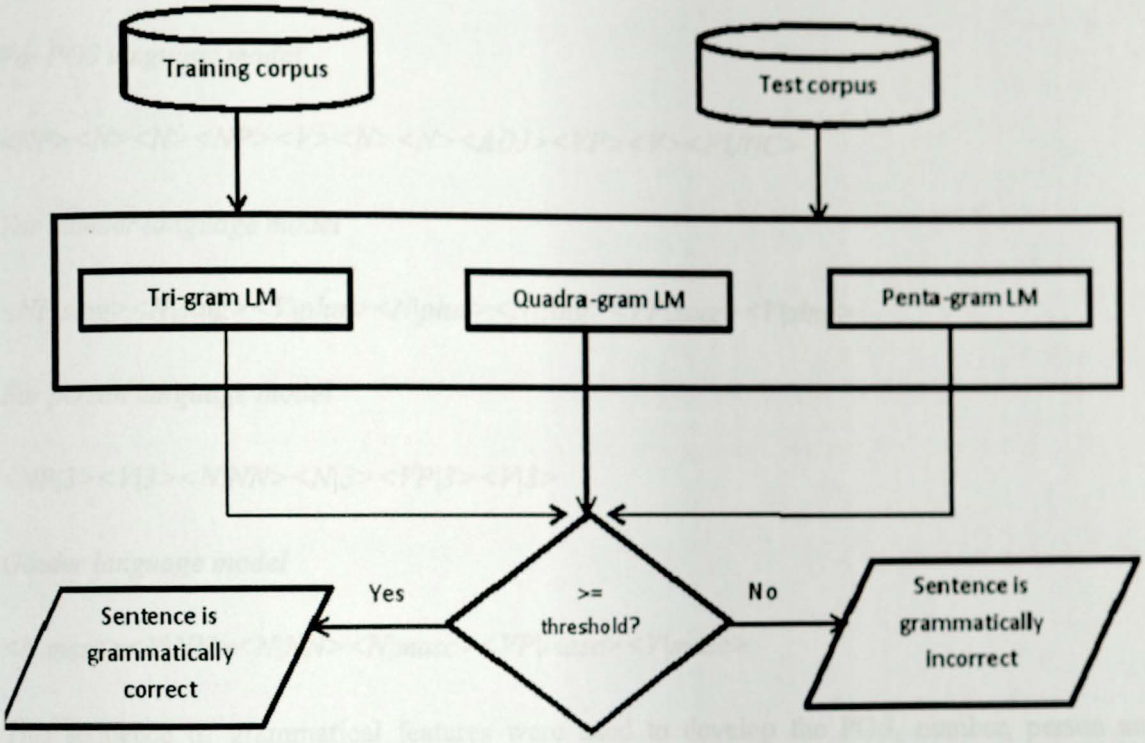


Figure 3 Architecture of N-gram Language model

5.2.2. Language models

The LM model is built from 3-gram up to 5-gram using SRILM toolkit. We built n-gram (n=3, 4, 5) language model of POS tags sequence along with their grammatical tags. We have built four language models, for POS tag sequence, POS tag with number marker, POS tag with person marker, and POS tags with gender marker. The grammatical feature marker is associated with POS tag mainly because the agreement depends on the POS tag class to have a specific

corpus.

Example:

በደሀት<NP> ቅነሳው<N|sing|3|masc> ሰኔድ<N> የመንግሲታዊ<NP> ያልሆኑ<V|plur|3|NN> ድርጅቶች<N|plur|NN|NN> ተሳትፎ<N|sing|3|masc> ከፍተኛ<ADJ> እንደሆነ<VP|sing|3|masc> ተጠቁመ<V|sing|3|masc>::<PUNC>

For POS language model

<NP><N><N><NP><V><N><N><ADJ><VP><V><PUNC>

For number language model

<NP|sing><N|sing><V|plur><N|plur><N|sing><VP|sing><V|sing>

For person language model

<NP|3><V|3><N|NN><N|3><VP|3><V|3>

Gender language model

<N|masc><V|NN><N|NN><N|masc><VP|masc><V|masc>

This sequence of grammatical features were used to develop the POS, number, person and gender LM for tri-, quadra- and penta-gram LM. The input sentence is assigned with probability to determine whether the sentence is grammatically correct or not. Sentence is said to be valid if the sequence probability is greater than the threshold value and invalid otherwise. Threshold value is selected by taking numerical points from 0-1 and the value which best classify the sentence is taken as threshold value. In this approach the selected points are computed (i.e. 0.1, 0.01, 0.001) and 0.01 is taken as the threshold value.

The test set which contain the POS, number, person and gender feature are passed to the tri-, quadra- and penta-gram LM.

PLUR|3|NN> ፣<PUNC> በርማ<N> ላይ<PREP> ተገኝተው<VP|PLUR|3|NN> ፣<P UNC> ወደቤታቸው<NP|PLUR|3|NN> ተመለሱ<V|PLUR|3|NN> ።<PUNC>

Tri-gram LM results of the following results:

> ***POS sequence***: *N VP PUNC N PREP VP PUNC NP V PUNC* = 0.64
>
> ***Number sequence:*** *VP|PLUR VP|PLUR NP|PLUR V|PLUR* = 0.09
>
> ***Person sequence:*** *VP|3 VP|3 NP|3 V|3*=0.16
>
> ***Gender sequence:*** *VP|NN VP|NN NP|NN V|NN*= 0.08

Each of the sequences is computed with the assigned threshold value (i.e. 0.01). The output of the tri-language model results 0.64, 0.09, 0.16 and 0.08 for POS, number, person and gender sequence respectively thus each of the probability is greater than the threshold value. Therefore the sentence is grammatically correct in each of morpho-syntactic feature.

5.3.1. Architecture of DP for grammar Checking

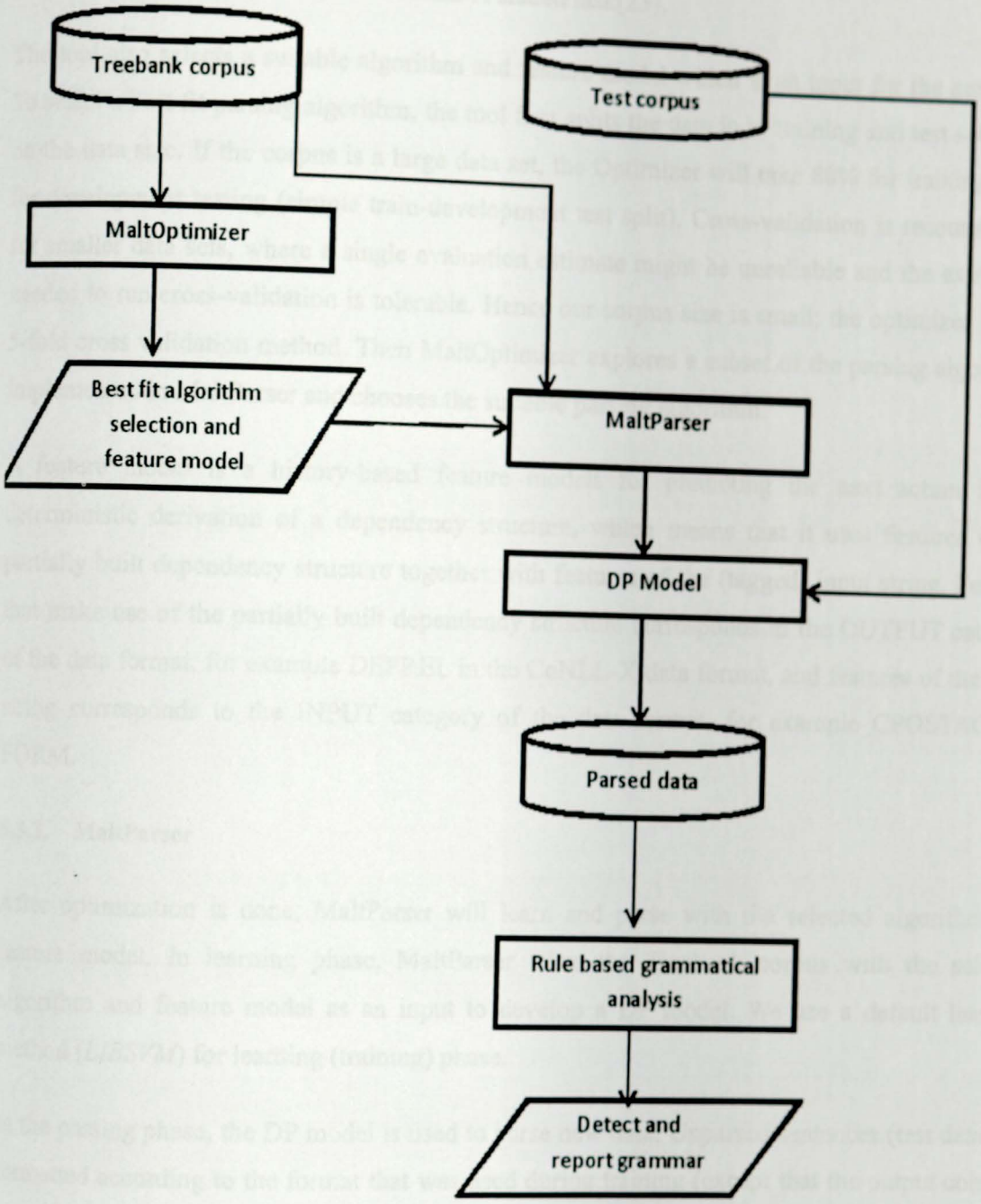


Figure 4 Architecture of DP for grammar Checking

The preprocessed Treebank corpus is passed to MaltOptimizer tool for data validation. Data validation is done by checking the Treebank corpus is in appropriate CoNLL-X format by using official validation script from the CoNLL-X shared task[23].

The tool also selects a suitable algorithm and feature model which is an input for the next step. To select a best fit parsing algorithm, the tool first splits the data in to training and test set based on the data size. If the corpus is a large data set, the Optimizer will take 80% for training, 20% for development testing (simple train-development test split). Cross-validation is recommended for smaller data sets, where a single evaluation estimate might be unreliable and the extra time needed to run cross-validation is tolerable. Hence our corpus size is small; the optimizer selects 5-fold cross validation method. Then MaltOptimizer explores a subset of the parsing algorithms implemented in MaltParser and chooses the suitable parsing algorithm.

A feature model is a history-based feature models for predicting the next action in the deterministic derivation of a dependency structure, which means that it uses features of the partially built dependency structure together with features of the (tagged) input string. Features that make use of the partially built dependency structure corresponds to the OUTPUT category of the data format, for example DEPREL in the CoNLL-X data format, and features of the input string corresponds to the INPUT category of the data format, for example CPOSTAG and FORM.

5.3.3. MaltParser

After optimization is done, MaltParser will learn and parse with the selected algorithm and feature model. In learning phase, MaltParser takes the Treebank corpus with the selected algorithm and feature model as an input to develop a DP model. We use a default learning method (*LIBSVM*) for learning (training) phase.

In the parsing phase, the DP model is used to parse new data. Unparsed sentences (test data) are formatted according to the format that was used during training (except that the output columns for HEAD and dependency relation (DEPREL) are missing). In our case, the first eight columns

parsed data.

5.3.4. Rule based grammatical analysis

Analysis of grammatical agreement will take place after the input data is parsed. The rules discussed in section 5.1.6 is used to analyze grammatical agreement between the words. Since the words are presented as functional and lexical form the grammatical module will merge in to their content words. The following example is the output of the parsing module from the test corpus.

Example

1	ልጁ ች	ልጁ ች	NOUN	NOUN	Number=Plur	4	Obj	_	_
2	ኡ	ኡ	DET	DET	_	1	Det	_	_
3	ን	ን	PART	ACC	_	1	Case	_	_
4	ጠር ተ	ጠር ተ	VERB	VERB	VerbForm=Conv	11	compound:svc	_	_
5	ሀ	ሀ	PRON	SUBJC	Gender=Masc|Number=Sing|Person=2	4	Nsubj	_	_
6	አህል	አህል	NOUN	NOUN	_	11	Obj	_	_
7	ኡ	ኡ	DET	DET	_	6	Det	_	_
8	ን	ን	PART	ACC	_	6	Case	_	_
9	አ	አ	ADP	ADP	Case=Loc	10	Case	_	_

11	አስገባ	አስገባ	VERB	VERB	Voice=Cau	0	Root	_	_
12	ኧ	ኧ	PRON	SUBJC	Gender=Masc|Number=Sing|Person=2	11	Nsubj	_	_
13	።	።	PUNCT	PUNCT	_	11	Punct	_	_

Table 6 example is the output of the parsing module

This sentence is presented in content word form along with the values as shown below. The FEATS value is also changed while merging the words. Here the FEATS values holds the subject, object and the indirect object grammatical feature marker separated by '||' respectively. If the word does not have a FEATS value, it will have '_' or underscore marker.

Example

1	ልጆቹን	ልጆች	NOUN	NOUN	*_||Number=Plur||_	4	obj	_	_
4	ጠርተው	ጠርተ	VERB	VERB	*Gender=Masc|Number=Sing|Person=2||_||_	11	nsubj	_	_
6	እህሉን	እህል	NOUN	NOUN	*_||_||_	11	obj	_	_
10	አጎተራ	አ	ADP	ADP	*_||_||_	11	obl	_	_
11	አስገባ	አስገባ	VERB	VERB	*Gender=Masc|Number=Sing|Person=2||_||_	0	root	_	_

Table 7 sentence is presented in content word form

The grammatical analyses module will then extract elements which are ID, FEATS, HEAD, and DEPREL that are important to analyze the agreement between words. These features are taken as important element because ID is a unique value to identify the word in the sentence, FEATS holds the morpho- syntactic features of the word and HEAD convey the dependency relation between the words and DEPREL holds the syntactic information of the word.

The grammatical analyses module starts by analyzing the first word in the sentence. From the above example the first word has ID = 1, FEATS = *_||Number=Plur||_, HEAD = 4 and DEPREL = obj values.

The module identifies the dependency between the words by looking the value of HEAD. The current word has a HEAD value of 4, which means the word is dependent on the word with ID number 4. Therefore the two dependent words must agree on the FEATS value (i.e. the grammatical agreement).

1 _||Number=Plur||_, 4 obj

4 *Gender=Masc|Number=Sing|Person=2||_||_ 11 nsubj

The grammatical analyses module takes subject marker of both words and analyze whether they possesses the same agreement according to the language behavior. The object and indirect object will also be analyzed if the words have grammatical feature marker which indicate the object and indirect object. The subject, object and indirect object marker of the words must match in gender, person and number values.

The ID = 1 word does not have a subject and indirect object marker in which it is represented as '_' but has an object marker which is Number=Plural whereas the ID = 4 word has a subject marker Gender=Masculine, Number=Singular and Person=2 but does not possesses object and indirect object marker. Hence DEPREL value indicates the words are object and subject of a sentence respectively. Notice that the subject of a sentence is a verb which holds a subject

the system will assume that the sentence is grammatical correct. Therefore the two dependent words are grammatical correct.

The module will then continue to analyze the second word. The second word has ID = 4, FEATS = *Gender=Masc|Number=Sing|Person=2|_|_, HEAD = 11 and DEPREL = nsubj. The ID = 11 is extracted because the HEAD of the word indicated the dependency relation exit between them

4 *Gender=Masc|Number=Sing|Person=2|_|_ 11 nsubj _ _

11 *Gender=Masc|Number=Sing|Person=2|_|_ 0 root _ _

The subject, object and indirect object markers are identified. The word with ID =4 has subject marker of Gender=Masculine, Number=Singular and Person=2 but it does not have object and indirect object markers hence it is represented as '_'. whereas, ID = 11 has a Gender=Masculine, Number=Singular and Person=2 subject marker but does not possesses object and indirect object markers. DEPREL indicates the ID = 4 is subject of a sentence and ID =11 is the root (main verb) of the sentence.

The gender, number and person marker of both words matches with ID number 4. Therefore the sentence is grammatically correct. The system will continue to check each word in the sentence until it reaches at the end of the sentence with the same procedure mentioned above.

approaches is presented. Probabilistic with rule based approach, higher order n-gram LM and DP are presented in this chapter respectively.

6.1. Probabilistic with rule based approach

As presented in the architecture in section 5.1.1 each of the modules are evaluated and presented as follows.

6.1.1. HMM POS tagger

8000 POS tagged corpus was classified into two sets, the training, and test sets. The training data contains 90% of the total corpus and the test set contains 10% of the total corpus. The POS tagger was trained by using two algorithms i.e. HMM and TnT. The results are shown below.

Algorithms	Accuracy
HMM algorithm (without smoothing technique)	79.9%
HMM algorithm (Lidstone smoothing technique)	81%
TnT algorithm	69.2%

Table 8 POS tagger evaluation

As the above table shows better performance is achieved by HMM algorithm with lidstone smoothing technique therefore HMM algorithm with smoothing is used as a POS tagging algorithm for this approach.

6.1.2. TnT Chunker module

509 sentences were manually chunk tagged in which 10% of the chunked data is used for testing. By comparing the result of the HMM and TnT algorithms, the one which performs better is taken

of accuracy.

6.1.3. Feature Extraction

We evaluate SOV extraction module manually. Extraction is evaluated by how many times the system correctly identified the subject, object, adjective, adverb, and verb of a sentence. If the feature extraction is not correctly identified then all the succeeding actions will be invalid. Any word class is incorrectly extracted by this module then the subsequent module analyzes the wrong words.

		Precision (%)	Recall (%)	Accuracy (%)
Detection	Adjective	81	54	88
	Subject	91	88	63
	Object	70	88	89
	Adverb	62	50	86
	Verb	100	100	100
Total				**85.8**

Table 9 feature extraction

Incorrect feature extraction can occur for different reasons. As we observed from our experiment, for instance, an adjective is detected wrongly mainly because of wrong POS tags. Adverb also fails to be correctly detected due to incorrect POS tags. If we look into subject detection 8% of the subject detection fails because of wrong chunking. The phrase in a sentence is incorrectly chunked that will lead to detect the wrong subject.

6.1.4. SOV Sequence checking module

SOV word order is manually evaluated. Here our assumption is subject, object and verb of a sentence always come one after the other. Even though OSV is a valid sentence in Amharic language [35]. 100 correct SOV order sentences were given to SOV sequence checker and 8

a sentence. The accuracy of SOV sequence checking module scores 79%.

Example: ውድድሩንN B-NP_C ያሸነፈውVP I-NP_C ተማሪN I-NP_C ሽልማትN O ሊቀበልVN O መጣV O ።PUNC *O. this sentence was detected as incorrect word sequence because an object of the sentence was wrongly detected or analyzed. The system detects SOV as follows:*

Subject: ተማሪ ; *Object:* ውድድሩን ; *Verb:* መጣ

The object 'ውድድሩን'precedes the subject 'ተማሪ', as our assumption, the sentence is said to be a valid sentence if it has S-O-V word order. Therefore in this example, the sentence is invalid because it has an OSV sequence.

6.1.5. Agreement module

The agreement and suggestion module is evaluated manually by checking each agreement and calculating the precision, recall and accuracy metrics.

Agreement	Precision (%)	Recall (%)	Accuracy (%)
Adjective – subject	69	25	79
Subject-verb	94	64	63
Object – verb	81	70	86
Adverb – verb	86	41	78
Total			76.5

Table 10 evaluating Agreement module

Note that HornMorpho analyzes only noun and verb of Amharic language [42]. In order to analyze adjectives with HornMorpho, we assumed all adjectives are as a noun. We conduct a simple experiment to evaluate the performance of HornMorpho on adjective analysis by taking all adjectives from the WIC corpus. The performance of the analyzer fails to analyze 18.9% of the words because of the duplicative behavior of adjectives and the analyzer does not know the

are derived from nouns.

76.5% of overall accuracy is achieved whereas 23.5% fails in grammatical analysis due to wrong detection in feature extraction phase or wrong analysis of the words by HornMorpho.

6.1.6. Suggestion

If the agreement module incorrectly identified the agreement then the suggestion module led to generate an incorrect answer. If incorrect detections occur and pass to this module, the suggestion of the given incorrectly detected word is not counted as correct generation, even if the generated word is correct. The result of the suggestion module is discussed in the following table.

Suggestion	Precision (%)	Recall (%)	Accuracy (%)
Adjective - subject	40	8	57
Subject-verb	27	21	47
Object – verb	44	4	77
Adverb – verb	44	44	60
Total			61

Table 11 evaluating suggestion module

The suggestion fails to generate the correct output because HornMorpho generator does not know the given root -feature combination and generator does not yield the correct output for the given feature.

In general, the whole process (this approach) performs 61% of accuracy while the remaining fails because the feature extraction module fails to extract the correct word, the feature extraction module fails mainly because of wrong POS tag therefore We further demonstrate the effect of POS tag in grammar checker by gold standard POS tagged data as input for the grammar checker on the following section.

POS is a very important input for a machine learning process, We have tried to conduct this experiment by including POS tagging in the preprocessing phase. The corpus does not need any tagger module because we are using the manually POS tagged corpus as an input to the chunker module. We used the test set that we already prepared (section 4.3.1) and each of the words is manually POS tagged.

6.2.1. Chunker module

The chunker module which is trained by TnT is used to chunk the test sentence. The output will be passed to the feature extraction. The sentence will be checked whether it is in a valid sequence by SOV sequence checker. If it is a valid sentence, the agreement module takes place. As mentioned in the above section, the agreement module will use HornMorhpo for analysis and generation for suggestion module. This experiment is done to compare the effect of POS tagger in grammar checking system using gold standard corpus.

6.2.2. Feature Extraction and SOV sequence checker

The output of the chunker module is passed to the extraction phase to extract the subject, object, verb, adverb, and the adjective of the given sentences. The following table represents the output of the experiment of each feature with SOV sequence checker.

Detection	Precision (%)	Recall (%)	Accuracy (%)
Adjective	95	81	96
Subject	100	70	70
Verb	100	100	100
Object	76	94	92
Adverb	100	96	99
Total			91

Table 12 evaluation of word class detection

As the experiment shows, the detection system performs better than the previous experiment. For instance, the adverb of sentence detection increases by 13%. The system detects the adverb when the system identifies a word with an adverb tag within a sentence. Hence the POS tag is correct the system detects all adverbs that exist in the test sentence. The verb of a sentence is correct because based on our assumption the verb always exists at the end of a sentence with verb class POS tags. Therefore, the system directly goes to the end of a sentence to extract the verb of a sentence.

Detecting object of the sentence improves insignificantly comparing with adverb and subject of the sentence. This is because of the complex nature of the object in the Amharic language. As mentioned in previous sections, the system only takes a word as an object if it has an accusative marker and direct object of the sentence. But accusative case marking is not necessary/ sufficient condition for a word to be taken as object marker [46]. The object marker can be analyzed as double clitic even though clitic doubling is debate many researchers and history of Amharic literature on object marker [46].

The SOV sequence module is dependent on object extraction and chunker output. If the object is not detected correctly, then there is a great chance that the sentence is invalid. The SOV extraction performance scores 87%. The subject detection performance rises from 63% to 70% accuracy. In most cases, the subject was wrongly detected because the chunker module did not chunk correctly.

Hence, the overall accuracy of the system increases significantly. If feature extraction is correctly identified, the agreement analysis performance also will increase and if extraction fails to correctly identify then the agreement analysis performance will decrease.

The agreement of adjective – subject, subject-verb, object – verb, and adverb –verb is discussed as follows.

Agreement	Precision (%)	Recall (%)	Accuracy (%)
Adjective – subject	89	73	
Subject-verb	96	60	60
Object – verb	86	91	
Adverb – verb	93	45	86
Total			84

Table 13 Evaluation of agreement module

Adjective and subject agreement analysis increases from 79 to 94% of accuracy because of correct detection of subject and adjectives. The other reason for better performance is the correct output of HornMorpho analyzer.

Adjective and subject agreement in number, person, and gender feature marker is analyzed and if each of the feature markers is matched then it is considered as valid otherwise invalid

Subject and verb agreement do not make a significant change. The main reason is that in both experiments the verb detected correctly with the precision of 100% since the verb of a sentence appears at the end of every sentence and the system looks the verb at the end of the sentence (refer section 5.1.4.).

The upsurge of object and verb agreement is 8% of accuracy. The verb might not hold an object feature marker or hold the feature marker of the indirect object in a sentence. When this exception occurs the object and verb agreement will be taken as valid. This might not hold true for all sentences, therefore, further investigation is needed for the object and verb agreement. [46] explains this effect by giving the following example.

Aster doro-wa-n lä-Girma sät't'-ätʃtʃ-iw

Aster chicken-DEF.F-ACC to-Girma give.PF-3FS.S-3MS.O

'Aster gave the hen to Girma.'

In the above sentence, "lä-Girma" is an indirect object of the sentence but does not possess accusative agreement with it. But the accusative feature is with the word "doro-wa-n" which is the direct object. However, the verb-object marker references the indirect object Girma (third-person masculine singular) and not 'the hen' (third-person feminine singular).

HornMopho analyzer outputs incorrect agreement feature for object and verb due to incorrect identification of the root/stem of verb and noun respectively. And for some of the words, the analyzer does not know the word grammatical feature marker.

The adverb and verb detection increase with a precision of 8%. The adverb analysis is done using a bigram model since HornMorpho does not analyze adverb[42]. The immediate verb after the adverb is taken and analyzed. The higher probability of adverb will be assigned to the adverb tense marker and the verb will be analyzed with HornMorpho and if the adverb and verb match on the tense form then the sentence is correct tense feature marker. Otherwise, the sentence will be considered as incorrect and passed to suggestion phase. Better performance can be achieved by increasing the vocabulary of adverb hence we extract the adverbs from the corpus; the vocabulary is very much small.

6.2.4. Suggestion

The suggestion module takes place if the sentence fails to match in agreement. The module will take the word in the root or stem form with an appropriate feature of the word. The suggestion is given for grammatically incorrect word.

The suggestion performs 72% of accuracy in which it generates almost all grammatically incorrect words. Since the Amharic language is very inflective, it might fail to get the exact word

The result showed a significant increase in the performance on detecting grammatically invalid sentence using gold standard POS tagged data. Therefore POS tag has a significant effect in identifying syntactically in/valid sentences. Therefore a high quality of automatic POS tagger is needed for error detection in grammar checker

6.3. N-gram language model based grammar checker Evaluation

The prepared test data mentioned in section 4.3.1.5 which contains 150 sentences, out of which 80 sentences are grammatically incorrect sentences and contains all error types (errors occur in number, gender and person). The remaining sentences are grammatically correct sentences. If the probability of a sentence is less than the given threshold value (0.01), it is considered as grammatically incorrect sentences.

The error coverage of tri-, quadra- and penta- gram LM is computed and presented in column label 'correctly detected' and 'Incorrectly detected'. 'Correctly detected' column represent the valid and invalid sentence detected by the LM which are in fact valid or invalid. The invalid sentences which are detected as correct sentences or vice versa is represented in 'Incorrect detection' column.

Tri-gram LM

	Correctly detected	Incorrectly detected	Accuracy
Agreement in Number	53	97	32.5%
Agreement in gender	121	29	85%
Agreement in person	118	32	83.7%

Table 14 Evaluation of Tri-gram LM

is better than number agreement because of false positive detection of the system was high.

Quadra-gram LM

	Correctly detected	Incorrectly detected	Accuracy
Agreement in Number	42	87	[illegible]
Agreement in gender	105	24	81.3%
Agreement in person	108	22	[illegible]

Table 15 evaluation of Quadra-gram LM

In quadra-gram LM, the result is similar to the trigram LM. The detection of false positive is high and

The error coverage of person and gender agreement increase compared to the tri-gram LM. Due to the rise of false positive quadra-gram LM result in detection number agreement did not show performance improvement.

Penta-gram LM

	Correctly detected	Incorrectly detected	Accuracy
Agreement in Number	49	80	37.9%
Agreement in gender	111	18	86%
Agreement in person	109	20	84.4%

Table 16 evaluation of Penta-gram LM

performance. Pentagram LM did not show a significant increase in detection of all error types occurred in a sentence because the language model does not cover all the agreement in Amharic language and the sequence of feature markers are rare in the case of the pentagram language model. In addition the corpus used to develop the 5-gram LM is small that makes the LM suffer from data sparcity.

6.4. DP evaluation

For evaluation, MaltEval toolkit is used. This tool holds LAS, UAS and LA measurement metrics.

6.4.1. Algorithm selection

	LAS
NivreEager algorithm	77.5
NiverStandaed	77.54
StackProjective with pseudo-projective	77.64
NiverStandard with pseudo-projective	77.81

Table 17 evaluation of DP

The table shows NivreEager algorithm shows the LAS improvement from the baseline i.e. 75.08% with 2.019% but the NiverStandard with pseudo-projective algorithm shows 2.73% improvement over the baseline. Therefore NiverStandard with pseudo-projective algorithm is the best fit for the data.

The output of MaltParser (DP model) is evaluated and presented as follows.

	LAS	UAS	LA	Overall accuracy
Grammatically correct sentences	76.2	92.1	80	79.7
Grammatically incorrect sentences	86.8	96.9	88.5	87.4

Table 18 evaluation of MaltParser

6.4.3. Grammatical analysis

Grammatical analyses of input sentences are evaluated using accuracy measurement metrics. The accuracy of correct and incorrect sentences is presented in the following table.

	Accuracy
Grammatically correct sentences	86.04%
Grammatically incorrect sentences	81.3%
Overall	84.7%

Table 19 evaluation of grammatical agreement of DP

As shown in the table 18.7% of grammatically incorrect sentences were detected as valid sentence. This is because the parser fails to annotate the words with correct dependency label, whereas 15.3% of correct sentences were detected as grammatically incorrect sentences.

In general, considering the test set the following result was obtained:

- 70.4% of error analysis was correctly detected by DP whereas 18% of the test sentence is correctly detected by only DP.
- 60% of error analysis was correctly detected by probabilistic method with rule-based approach and 11% of error detection is achieved by only this approach
- 47.7% of grammatical error detection was correctly identified by higher order n-gram and 9% of the test data is recognized only by this approach.

Probabilistic with rule based approach is conducted to check the grammatically correctness of Amharic sentence. Rather than conducting on rule-based and/or statistical method independently, probabilistic with rule based approach of the two approaches surpass on detecting a syntax error in the Amharic language.

The second experiment uses tri-gram, Quadra-gram, and penta-gram language model to check long distance grammatical correctness of a sentence. The language model is built for POS tag, number tag, person tag, and gender tag. For a given sentences with an appropriate word tag, the system calculates the probability of each of the tag from the corresponding language model. For morphologically rich language like Amharic, large data set is needed to train LM as the result higher order n-gram LM did not show a significant increase on the performance of the system comparing to the 3-gram LM.

The third experiment is conducted to handle grammatical agreement by using DP. This experiment has two phases. The first phase is parsing the input sentence using the DP model and the second is grammatical analysis phase. As of the finding, DP is suitable in handling long distance agreement in a sentence. Hence DP uses a label (HEAD) to annotate the dependency exits among the words. Based on the assumption if a word is dependent on the other, then the two words should agree on grammatical features. Moreover the DP convey a grammatical information (DEPREL) i.e. Dependency relation to the HEADwhich indicates whether the word is an object, subject or indirect object of the sentence this label helps to handle the free word-order of Amharic sentences. The DP approach surpass both probabilistic with the rule-based approach and higher order n-gram LM in handling long distance agreement with in a sentence and free word order.

7.2. Recommendation

Based on the findings and limitation of this research the following lists are recommended for future works for researchers interested in this domain area.

to develop better data to implement the system.

- **Object – verb agreement** is a challenge in detecting a correct object in a sentence. The behavior object is somehow challenging in detecting the correct word in a sentence. Objects in the language may or may have not an indicated morpheme. Furthermore, the indicator morphemes might not be sufficient to take the word as an object. To check the agreement of an object and a verb the verb needs to have an object marker. But there are verbs which do not have an object feature marker. In our system (probabilistic method with rule-based approach) when this exception occurs, the system will take the sentence as a correct agreement between the object and verb. But this might not work for all sentences in the language. Therefore, we recommend investigating more about the object and verb behavior.

- **Adjectives** are derived from nouns, stems or verbal roots by adding suffixes. Adjectives have a reduplication of consonant to indicate number feature marker. The reduplication behavior of adjectives needs to be addressed in how to analyze and/ or generate a correct form of the adjective word. This research work did not address the repetitive behavior of adjectives. We recommend considering this behavior of adjective on the future work.

- Analysis of **adverb** should be handled in a certain way that better result to be achieved rather than extracting a list of adverbs from a certain corpus. The reason is that the data derived from a corpus will give an unsatisfactory result.

thesis) Addis Ababa Univeristy, 2013.

[2] A. M. Gezmu, B. E. Seyoum, and A. Nürnberger, "Contemporary Amharic Corpus : Automatically Morpho-Syntactically Tagged Amharic Corpus," in *Proceedings of the First Workshop on Linguistic Resources for Natural Language Processing*, 2018, pp. 65–70.

[3] B. E. Seyoum, Y. Miyao, and B. Y. Mekonnen, "Morpho-syntactically Annotated Amharic Treebank," 2008.

[4] Daniel, H. James, and Martin, "Part-of-Speech Tagging," *Speech, Lang. Process.*, no. Chapter 12, 2016.

[5] M. Kassa Gobena, "Implementing an open source Amharic resource grammar in GF," no. November, 2010.

[6] A. D. Mohammed, "A top-down chart parser for amharic sentences," 2015.

[7] V. Henrich and T. Reuter, "LISGrammarChecker : Language Independent Statistical Grammar Checking," 2009.

[8] D. Naber, P. F. Kummert, T. Fakultät, and A. Witt, "A Rule-Based Style and Grammar Checker," 2003.

[9] K. Hagen, J. B. Johannessen, and P. Lane, "Some problems related to the development of a grammar checker," no. 1, 2001.

[10] M. K. Gobena, "Implementing an open source Amharic Ressource Grammr in GF," 2010.

[11] N. S. Bhirud, R. P. Bhavsar, and B. V Pawar, "Grammar checkers for natural languages : a review," vol. 6, no. 4, 2017.

[12] A. Arppe, "Developing a grammar checker for Swedish," *Proc. NODALIDA*, pp. 13–27, 2000.

[13] M. S. Gill, G. S. Lehal, and S. S. Joshi, "A Punjabi Grammar Checker," pp. 940–944, 2008.

[14] J. Alam, N. Uzzaman, and M. Khan, "N-gram based Statistical Grammar Checker for Bangla and English."

[16] C.H.DAWKINS, *The Fundamentals of Amharic*. University of Virginia 1969.

[17] Atelach Alemu Argaw, "Automatic sentence parsing for amharic text: an experiment using probabilistic context free grammars," 2002.

[18] Daniel Gochel Agonafer, "An integrated approach to automatic complex sentence parsing for amharic text," 2003.

[19] Abeba Ibrahim, "A hybrid approach to amharic base phrase chunking and parsing," no. March, 2013.

[20] P. Hartwell, "Grammar, Grammars, and the Teaching of Grammar," vol. 47, no. 2, pp. 105–127, 2009.

[21] K. F. Shaalan, "Arabic GramCheck : a grammar checker for Arabic," no. April 2004, pp. 643–665, 2005.

[22] A. Zouaq, "Shallow and Deep Natural Language Processing for Ontology Learning : Learning : A Quick Overview."

[23] M. Ballesteros and J. Nivre, "MaltOptimizer : A System for MaltParser Optimization," no. 2006, pp. 2757–2763, 2009.

[24] H. K. Elnajjar, "Improving Dependency Parsing of Verbal Arabic Sentences Using Semantic Features" 2016.

[25] J. Nilsson, "User Guide for MaltEval 1 . 0 (beta)," vol. 0, pp. 1–33, 2014.

[26] J. T. Goodman, "A bit of progress in language modeling," *Mach. Learn. Appl. Stat. Group, Microsoft Res. One Microsoft Way, Redmond, WA 98052, U.S.A.*, pp. 403–434, 2001.

[27] D. Jurafsky and J. H. Martin, *Speech and Language Processing: An introduction to speech recoginition, computitional lingustics and natural language processing*. 2007.

[28] U. Singh and V. Goyal, "Punjabi Pos Tagger : Rule Based and HMM," *Int. Journals Adv. Res. Comput. Sci. Softw. Eng.*, no. 7, pp. 193–205, 2017.

[30] Brants Thorsten, "TnT - A Statistical Part-of-Speech-Tagger," *proceeding thee Sixth Appl. Nat. Lang. Proceessing Conf. ANLP-2000*, 2000.

[31] S. Renals, "Part-of-speech tagging (3) Recall : HMM PoS tagging Viterbi decoding Trigram PoS tagging Summary," no. October 2006, 2006.

[32] N. S. Bhirud, R. P. Bhavsar, and B. V Pawar, "A SURVEY OF GRAMMAR CHECKERS FOR NATURAL LANGUAGES," no. Cl, pp. 51–62, 2017.

[33] B. K. Bal, P. Shrestha, and M. P. Pustakalaya, "Architectural and System Design of the Nepali Grammar Checker," 2007.

[34] M. Miłkowski, "Automating rule generation for grammar checkers."

[35] B. Yimam, "yäamarIña säwasäw. EMPDE, Addis Ababa, 2nd. ed. edition," 2000.

[36] S. Wintner, "Morphological Processing of Semitic Languages," pp. 43–67, 2014.

[37] Y. Assabie, "Lecture 03 : Syntax and Parsing:[PowerPoint Presentation]," no. Cosc 709. 2014.

[38] C. W. Isenberg, *Grammar of the Amharic language*. London: Ann Arbor, 1842.

[39] M. A. Wondem, *The Syntax of Non-verbal Predication in Amharic and Geez*. LOT Trans 10 3512 JK Utrecht The Netherlands phone:, 2014.

[40] O. K. J. Carlberge r, R. Dome i j, V. Kann, "The Development and Performance of a Grammar Checker for Swedish: A language engineering perspective," no. 1987, pp. 1–17, 2004.

[41] M. Ballesteros, "Introduction to Dependency Syntax and Dependency Parsing," vol. 1, no. 31.

[42] M. Gasser, "HornMorpho : A system for morphological processing ofAmharic, Oromo, and Tigrinya," no. May, pp. 2–5, 2011.

[43] A. Stolcke, S. R. I. International, and M. Park, "SRILM — An extensible language modeling toolkit."

[44] J. Hall and J. Nilsson, "MaltParser : A Data-Driven Parser-Generator for Dependency Parsing," pp. 2216–2219.

[45] G. D. Little, "Word order function typology: The Amharic connection," 1977.

[46] R. Kramer, "Object Markers in Amharic *," *georgetown university annual conference on african linguistics*, pp. 1–11, 2010.

Appendix

A. POS tag with their description

PoS tags	Description
[illegible]	Adjective
ADJC	Adjective with conjunction
[illegible]	Adjectival phrase
ADJPC	Adjectival phrase with conjunction
[illegible]	Adverb
AUX	Auxiliary
[illegible]	conjunction
INT	Interjection
[illegible]	Noun
NC	Noun with conjunction
[illegible]	Noun phrase
NPC	Noun phrase with conjunction
[illegible]	Cardinal Number
NUMOR	Ordinal Number
[illegible]	Number with phrase
NUMPC	Number phrase with conjunction
[illegible]	Preposition

PREPC	Preposition with conjunction
PRONC	Pronoun with conjunction
PRONPC	Pronoun phrase with conjunction
PROPN	Proper Noun
[illegible]	Punctuation
UNC	Unknown or unclear
[illegible]	Verb
VC	Verb with conjunction
VN	Verbal noun
VP	Verb phrase
[illegible]	Verb phrase with conjunction
VREL	Relative verb

Table 20 List of Part of speech Tag with their description

B. Chunk tag with their description

Chunk type	Description
[illegible]	Beginning of [illegible]
I-NP_C	Inside noun phrase chunk
[illegible]	Beginning of verb phrase chunk

I-VP_C	Inside of verb phrase chunk
[redacted]	
I-PP_C	Inside of a prepositional phrase chunk
[redacted]	

Table 21 List of chunk tag and their description

C. Possible values for the dependency relation label.

Field number	Field name	Description
[redacted]	ID	Token counter, starting at 1 for each [redacted]
2	FORM	Word form or punctuation symbol.
[redacted]	LEMMA	Lemma or stem (depending on particular data set) of word form, or an underscore if not available.
4	CPOSTAG	4 Coarse-grained part-of-speech tag, where tag set depends on the language
[redacted]	POSTAG	Fine-grained part-of-speech tag, where the tag set depends on the language, or identical to the coarse-grained part-of-speech tag if not available.
6	FEATS	Unordered set of syntactic and/or morphological features (depending on the particular language), separated by a vertical bar (|), or an

underscore if not available.

7	HEAD	[illegible]
8	DEPREL	Dependency relation to the HEAD. The set of dependency relations depends on the particular language. Note that depending on the original treebank annotation, the dependency relation may be meaningful or simply 'ROOT'.
9	PHEAD	Projective head of current token, which is either a value of ID or zero ('0'). For the underscore if not [illegible] on the original treebank annotation, [illegible] with ID of zero. The dependency structure resulting from the PHEAD column is guaranteed to be projective (but is not available for all languages), whereas the PANACEA Project's Platform for Automatic, Normalized Annotation and Cost-Effective Acquisition – Grant Agreement no. 248064 structures resulting from the HEAD column will be non-projective for some sentences of some languages (but is always available).
10	PDEPREL	Dependency relation to the PHEAD, or an underscore if not available.

The set of dependency relations depends on the particular language. Note that depending on the original Treebank annotation, the dependency relation may be meaningful or simply 'ROOT'

D. Sample of grammatically incorrect sentence

1. ትልቁዛፍበመጥረቢያዛሬተቆረጠች።
2. ውድድሩንያሸነፈውተማሪሽልማትልትቀበልመጣች።
3. ተማሪውደብተሩንቀደደችው።
4. አንበሳዋሲጠግብተንጠራራ።
5. ልጁትምህርቱንበደንብያጠናል፤ስለዚህትሩውጣትታመጣለች።
6. ልጅቷበጠዋትተነስታወደገበያሄደ።
7. ሴትየዋነገወደድሬድዋሊሄድነው።
8. የታመመውልጅድኖተነሰታልች።
9. ውድድሩንያሸንፈውነበዙተማሪቆንጆመፅሃፍተሸለመች።
10. ልጅቷደብትሩንይዞወደትምህርትቤትሄደ።
11. ሰውየውመፅሃፉንአንብቦሲጨርስወደቤቱሄደች።
12. ታታሪዋተማሪተሸለመ።
13. ፈተናውንያለፈችውተማሪደስታውንሊገልፅንመጣ።
14. እሷወጥታበዛውአልቀረም።

15. ነብዟተማሪወደትምህርትቤትአልመጣም

16. ትልቋሰውዬለችሎትከቀረበበኋላልጁንእንዲያሳድግተወሰነለት።

17. ልጆቹበጣምባልገ።

18. ሴቶቹሰከረዉወደቀ።

19. ትናንሾቹልጅአየተጫወቱነው።

20. እሱአየሮጠወደቤቱመጡ።

21. ልጆቹበህብረትትልቅዝማሬአሠማ።

22. አሷብዙመፅሀፍቶችገዙ።

23. መፅሃፉበስድስትክፍሎችተከፋፍለውቀርበዋል።

E. Sample of grammatically correct sentence

1. ኢማተልፈያስገነባቸው17 ትምህርትቤቶችአገልግሎትመስጠትጀመሩ።

2. ለከተሞችልማትአቅምግንባታፕሮግራምትኩረትይሰጣልተባለ።

3. ሁሉምከልሎችእንደየደረጃቸውየአቅምግንባታፕሮግራምተካፋይእንደሚሆኑምአቶጉተማተናግረዋል።

4. በቦረናየአህዴድተሃድሶውይይትተጀመረ።

5. በእንግሊዝየሚኖሩኦትዮጵያውያንየጋራመድረክመሰረቱ።

6. የሰራዊቱአቋምበአገሪቱየፖለቲካህይወትውስጥአዲስምእራፍመክፈቱተጠቆመ።

7. ትምህርትሚኒስቴርለኤችአይቪኤድስመከላከያ10.8ሚሊየንብርመደበ።

8. ሽብርተኝነትጸረ-ቱሪዝምበመሆኑድርጊቱንለመከላከልህብረተሰቡየበኩሉንድርሻእንዲወጣተጠየቀ።

9. ኮሜሳሰብልንከተባይናከበሽታለመከላከልየሚያስችልፕሮጀክትነደፈ።

10. በደንቆሮደንወስጥየቀይቀበሮቁጥርእየጨመረነው።

11. በሶማሌክልልካለፈውአመትብልጫያላቸውተማሪዎችአልፈዋል።

14. በድህንትቅነሳውሰነድየመንግስታዊያልሆኑድርጅቶችተሳትፎከፍተኛእንደሆኑተጠቆመ።

15. በአውደትናቴላይበኢትዮጵያየሚገኙመንግስታዊያልሆኑድርጅቶችበሙሉመሳተፋቸውንዋልታእንፎርሜሽንግለክልዘግቧል።

16. መገናኛብዙሃንለሚያጋልጧቸውበልሹአሰራሮችናአስተዳደራዊበደሎችመንግስትእርምጃእንዲወስድጋዜጠኞችጠየቁ።

17. 14 የአፈርምርመራላብራቶሪዎችግንባታተጠናቀቁ።

18. ሻእቢያተጨማሪከፍተኛባለስልጣናትንአሰረ።

19. ከሰባዊመብትተሟጋቾችውግዘትእየደረሰበትነው።

20. ኢማተልፈበሻእቢያወረራየወደሙተቋማትንለመገንባትእንቅስቃሴጀመረ።

21. ኤችአይቪኤድስንለመከላከልበእንቆቅልሾችናምሳሌያዊአነጋገሮችመጠቀምእንደሚያሻተጠቆመ።

22. በዞኑከ140ሺየሚበልጡከሻይረሱጋርየሚኖሩሰዎችእንዳሉከጤናመምሪያውየተገኘመረጃይጠቁማል።

23. ውድድሩበቀጥታቴሌቪዥንለአለምእንደሚሰራጭተጠቆመ።

24. ከኤርትራ205 ኢትዮጵያውያንወደአገራቸውተመለሱ።

25. ከእንስሳትከሚገኘውተረፈምርትበቀን180 ኩንታልጥሬእቃእንደሚዘጋጅተገለጸ።

26. ምክርቤቱከጣሊያንተቋምጋርበመተባበርየአቅምግንባታስልጠናአዘጋጀ።

27. ስልጠናውእስከየካቲትወርማብቂያድረስእንደሚቆይናበጣሊያንናበእንግሊዝኛቋንቋዎችእንደሚሰጥተናግረዋል።

28. በደቡብየዳኞችንስነምግባርናየሙያብቃትለማሻሻልፕሮጀክትተቀርጾእንቅስቃሴተጀመረ።

29. እንዲሁምለተለያዩየዞኖችናወረዳፍርድቤቶች55 የተለያዩየህግማጣቀሻመጽሀፍትእንደተሰራጩገልጸዋል።

30. የአውሮፓህብረትአምባሳደሮችበሻእቢያአፈናላይተቃውሙአሰሙ።

31. ለሆሊውድፊልምስራ255 ኢትዮጵያዊያንወደናሚቢያለጸዙነው።

32. ማእከሉሁለትየተሻሻሉዝርያዎችንበምርምርአገኘ።