**FINAL YEAR PROJECT REPORT**

A close up of a logo

Description automatically generated

SIDATH

**An Open Source Grammar Checker for Sinhala**

**Supervised By**

Prof. Gihan Dias

**Group 40**

Piyumal Dhananjaya 150119U

Kalindu Kumarasinghe 150345H

Prabhash Rathnayake 150529C

**25th of November 2019**

**Department of Computer Science and Engineering**

**Faculty of Engineering**

**University of Moratuwa**

This report is submitted in partial fulfillment of the requirements for the award of the degree of Bachelor of Science in Engineering at the University of Moratuwa, Sri Lanka.

# ABSTRACT

Project Title : Sidath (An Open Source Grammar Checker For Sinhala)

Authors : Piyumal Dhananjaya

Kalindu Kumarasinghe

Prabhash Rathnayake

Coordinator : Dr. Charith Chitraranjan

Supervisor : Prof. Gihan Dias

Sinhala is a rich and complex natural language. It was derived from two ancient Indo-Aryan languages, Pali and Sanskrit. In Sinhala there are two forms of language as written language and spoken language. The written language is richer with grammar. In formal writing, the correctness of many spelling and grammar components depends entirely on a set of clearly defined rules. The spelling and grammar are important to pass the correct meaning to the desired audience. When the world is progressing and the use of Sinhala language in electronic media is increasing it has become a big issue to spend time on manually proofread the content written in Sinhala language. There does not exist a Sinhala grammar checker yet. This might be due to the complexity of the Sinhala language.

There are good grammar checker tools which support major languages in the world, like English, Chinese and Arabic. But language handling tools for Sinhala is still in its infancy. There is no evidence of a fully working grammar check tool for Sinhala.

Sidath is an open source grammar checker for Sinhala which can be used as a stand-alone web application and as an API for software developers. It identifies grammar errors as well as provide suggestions to correct the error. Sidath is the solution for the problem of not having an open source grammar checker for Sinhala language. It is designed in a way that future users also can add many grammar rules without much difficulty. As the first phase, we developed a sophisticated Morphological Analyzer for Sinhala. Using this Morphological Analyzer, we can get an analysis about the words which can be used to check grammar when those words appear in a sentence.

We use MySQL database to keep processed datasets needed by the system.

People who deal with Sinhala documents will be benefited with this product.

**Keywords:** Morphological Analyzer, Sinhala Grammar Checker, POS tag, Finite State Transducer, Finite State Machine, Lexical Functional Grammar, Xerox Linguistic Environment

# ACKNOWLEDGEMENT

Sidath team sincerely acknowledges with so much pleasure and gratitude the guidance and supervision given by several persons throughout this project to make it a success. First, we would like to thank our supervisor Prof. Gihan Dias, senior lecturer at the Faculty of Engineering, University of Moratuwa.

We highly appreciate the utmost supportive hand lent to us by Dr. Charith Chitraranjan, our project coordinator. Also, we would like to convey our sincere gratitude to Dr. Surangika Ranathunga and Dr. Shantha Fernando who provided us with proper guidance and advice for our project. We would like to thank Mr. K Sarveswaran, Ms. Aloka Fernando, Ms. Nimasha Dilshani, Ms. Indu Herath and the members of the NLP Center - University of Moratuwa, for giving the much needed technical and linguistic support.

Our special thanks go to Prof. Miriam Butt, Ms. Jessica Zipf and Ms. Katerina Kalouli at University of Konstanz for the technical and linguistic support given to us specially regarding the Xerox Linguistic Environment and Computational Grammar.

Thanks to all the members of the academic staff and the non-academic staff of Computer Science & Engineering Department - University of Moratuwa, who helped us in various capacities to complete the project on time with satisfactory results.

Last but not the least we would like to mention all our colleagues of the CSE department, for not only helping us in giving the technical guidance at the times we needed, but also for working together as one family .We would also like to convey our appreciation to our parents and loved ones for the excellent support that they gave us in many ways.

# TABLE OF CONTENT

[**ABSTRACT**](#_heading=h.70iwdnjvyu4k)2

[ACKNOWLEDGEMENT](#_heading=h.30j0zll) 3

[TABLE OF CONTENT](#_heading=h.1fob9te) 4

[TABLE OF FIGURES](#_heading=h.3znysh7) **Error! Bookmark not defined.**

[INTRODUCTION](#_heading=h.3dy6vkm) 5

[**PROBLEM DEFINITION**](#_heading=h.5fa336acwwdm)6

[FUNCTIONAL SPECIFICATIONS](#_heading=h.1t3h5sf) 6

[LITERATURE REVIEW](#_heading=h.4d34og8) 7

[Research on Natural Language Processing](#_heading=h.2s8eyo1) 7

[Research on Similar Projects](#_heading=h.17dp8vu) 8

[Research on Sinhala Language](#_heading=h.3rdcrjn) 8

[Research on Existing Tools and Technologies](#_heading=h.lnxbz9) 14

[PROJECT WORKFLOW](#_heading=h.35nkun2) 16

[Background Study](#_heading=h.1ksv4uv) 16

[Grammar Rules Development](#_heading=h.44sinio) 17

[Morphological Analyzer for FOMA](#_heading=h.2jxsxqh) 21

[FUTURE WORK](#_heading=h.1y810tw) 36

[CONCLUSION](#_heading=h.2xcytpi) 37

[REFERENCES](#_heading=h.1ci93xb) 39

# INTRODUCTION

Sinhala is one of the official languages in Sri Lanka. It is the native language of Sinhalese who makes about 74 percent of the country’s population. Sinhala is used as a second language by minorities, making it the most widely spoken language in Sri Lanka. With the increased use of Information Technology, average Sri Lankans tend to use software to a significant extent in their day to day operations. A constraint has been the lack of software support for Sinhala language. Many people would prefer to do their own work in Sinhala using sophisticated software

Due to the high cost of commercial software many Sri Lankans have adapted the use of Free and Open Source software. For instance, the use of Open Office has increased. Open office has typing support for Sinhala. Recently some researchers have developed Sinhala spell checkers for open office. With a scrutiny we observed that these spell checkers also need to be developed further to increase their accuracy and efficiency. Still Sinhala language does not have a perfect spell checker. Other than spell checking, a Grammar checker for Sinhala has become a vital requirement in today's world.

Sidath Grammar Checker is useful for any user who deals with formal documents typed in Sinhala. Novelists, newspaper writers, bloggers who do their work in Sinhala, etc. can use this Sinhala Grammar Checker to check their work in real-time. Most of the government and non-government institutes use Sinhala for their official work. They can use our tool to type their documents and letters in Sinhala without any grammar error also they can use this tool to find and correct the grammar errors in existing digital documents typed in Sinhala.

Our Grammar checker API can be used by the developers to develop translation tools for Sinhala like English-Sinhala and Tamil-Sinhala translators. They can improve the quality of the translation mechanism by checking and correcting the grammar errors happen during the translation process.

Sidath Grammar Checker will be the first Sinhala grammar checking tool available for general internet users. This will enhance the use of Sinhala language in digital media. The development of the new tools like Sinhala translation tools will be motivated with the introduction of the Sinhala Grammar Checker.

# PROBLEM DEFINITION

A Grammar checker for Sinhala has become a vital requirement in today's world. There does not exist a Sinhala grammar checker yet. This might be due to the complexity of the Sinhala language as well as to the difficulty of designing a Grammar checker using the existing resources. All the application Programming Interfaces that have been used for grammar checking support Western languages.

# FUNCTIONAL SPECIFICATIONS

The final products of the project are as follows,

* Stand-alone web application to check Sinhala grammar.
* Sinhala verb morphology analyzer API.
* Sinhala Grammar Checker API for developers who are interested in developing more sophisticated tools for Sinhala like Sinhala translation tool.

We provide a simple word editor in our web application where the users can paste or type their Sinhala content on it. Then the application checks the sentences for their grammar mistakes. The system auto-corrects obvious mistakes and give suggestions if there is an ambiguity about the mistake. The OpenOffice users can install Sidath Grammar Checker as a plugin to OpenOffice writer.

Our grammar checker tool check for the following grammatical errors in sentences and give appropriate suggestions.

* Checking Subject-Verb agreement.
* Checking the use of correct Case.
* Proper use of Adjectives and Adverbs.

# LITERATURE REVIEW

The literature review was done under mainly under following categories.

1. Research on Natural Language processing
2. Research on similar projects
3. Research on Sinhala Language
4. Research on Lexical Grammar
5. Research on Finite State Transducers
6. Research on tools and technologies

## Research on Natural Language Processing

Natural Language Processing is discipline of Computer Science and linguistics. NLP can be summarized as the use of computation to understand Natural Language and speech [ref]. Computer science can be used to make computers understand and process languages and manipulate human language and speech patterns.

Natural Language Processing is a sophisticated inter-disciplinary combining high level of research of linguistics with computer science and mathematics. NLP helps to develop spell checkers, grammar checkers, part of speech tagger, and many more related to human languages. There are mainly two approaches for NLP

Machine Learning/Deep learning approach

Rule based/ Lexical Grammar approach

Data analytics/Deep learning uses data on the language to process [ref]. This need annotated data for processing. This is learning from past experience to learn the current situation. This needs a significant amount of data and computer power to create these models.

Rule based approaches [ref] are suitable for languages with well-defined languages (A complete set of grammar rules). The rules are defined as mathematical, regular or lexical grammar expressions and compiled to build a mathematical model to parse the language.

## Research on Similar Projects

DeepCorrection: Spell Correction and simple grammar correction [ref]

This project is using seq2seq model and deep learning. The training data includes real world mistakes in addition to the correct sentences. A clean text corpus of roughly 80 million sentences was built. It was tested on 30000 sentences of test data. But the real world application showed less accuracy. The model was tested on additional 1.4 million common grammatical mistakes. Out of 132 real world user’s input only 62 was corrected properly.

Urdu and the Parallel Grammar Project [1] research paper presents how to parse grammar rules for Urdu Language. Since both Urdu and Sinhala are Indo European languages and they follow almost the same structure. The research paper discusses the use of Lexical Functional Grammar for parsing grammar rules. This method is proved effective. The concept of morphology is discussed. Then lexicons are defined to create grammar rules. The lexicons are defined with parts of speech and grammar cases. Then C- structure and F structure for the grammar rules must be defined. Through this we can create grammar rules.

## Research on Sinhala Language

Structure of Sinhala Sinhala is a free word order language. Its unmarked word order is SOV; variant orders are also possible with discourse – pragmatic effects [ref]. A sentence can have all the possible orders of the main constituents with proper intonation. Figure 1

shows all the free-word order forms of the English sentence

A close up of text on a white background

Description automatically generated

*Figure 1: Free word order of Sinhala*

Sinhala is a head-final language, in which complements and modifiers appear before their heads

A screenshot of a cell phone

Description automatically generated

*Figure 2: Head-final property of Sinhala*

Traditionally, a sentence is divided into two parts; Noun Phrase (NP), and Verb Phrase (VP) [ref]. In Sinhala grammar, uktha (subject) and akyatha (predicate) are the two parts of a sentence. Subject and predicate in Sinhala sentences agree in number, gender and person. The studies of sentence structures of Sinhala have been made by several scholars. According to Abayasingha [ref] Sinhala has 25 types of simple sentence structures. However, in the present work, we have covered only the main sentence structures and a few complex structures. These are described in the following sections.

**Noun Phrase**

The Noun Phrase, denoted by NP, can be a common noun (N), pronoun (PrN) or a proper noun (PropN). In addition to the head noun, the Sinhala noun phrase consists of adjectival phrases and determiners. Sinhala NP has a very complex grammatical structure. It can consist of various clause structures, such as adjectival clauses, relative clauses, and subordinate clauses. Therefore, building a computational grammar, covering all the NP structures is complex. Figure 2 below shows the NP structure we have covered in the grammar developed in this work. An adjectival phrase (ADJP) is constructed with adjectives. According to Sinhala grammar, an adjectival phrase comes before the Noun (N) and after the Determiner (Det), if there is any determiner in the noun phrase. If the adjective is a qualitative adjective, then it can be constructed with Degrees (Deg) to intensify its meaning. In the traditional grammar of Sinhala, nama visheshana (adjectives) denote some quality or attribute of the noun. It can be divided into three classes, namely qualitative, quantitative and demonstrative [8]. However, in our grammar we do not consider the features of the ADJP. The words which denote the degree of the adjectives are added only before qualitative adjectives. i.e. the ADJP ‘ඉතා ෙහොඳ’ /it̪a: hon d̪ǝ/ (very good) is an adjectival phrase and it consists of a degree ‘ඉතා’ /it̪a:/ (very) and a qualitative adjective ‘ෙහොඳ’ /hon d̪ǝ/ (good), which appears before the adjective.

**Verb Phrase**

According to generative grammar, a Verb Phrase (VP) is a phrase headed by a verb. In addition to the verb, it consists of noun phrases and Adverbial Phrases (ADVP). The verb in Sinhala can be categorized as single verbs, compound verbs and auxiliary verbs. In this grammar we only consider single verbs. Generally, ADVP occurs before the verb in Sinhala sentences. However according to the features of the adverb, the position where the adverb occurs is decided. Figure 3 below shows the VP structure in Sinhala and what we have covered in the grammar. According to the structure, the verb appears in the final position. ADVPs may appear both before the verb and after the NP. If an adverb of manner occurs in the ADVP, the adverb can be combined with degrees to intensify the meaning. For example ‘ඉතා ෙගෙය’ /it̪a: ve:gǝyen/ (very fast) is an ADVP which appears as an adverb of manner.

**Cases in Sinhala Language**

In grammar, the case of a noun or pronoun is an inflectional form that indicates its grammatical function in a phrase, clause, or sentence. For example, a pronoun may play the role of subject ("I kicked the ball"), of direct object ("John kicked me"), or of possessor ("It is my ball"). – [ref] http://en.wikipedia.org/wiki/Grammatical\_case

This is another critical characteristic which can be used to define the POS category of a word. There are nine grammatical cases in Sinhala language. It can be illustrated with examples as follows.

Derivations of nouns as per cases Case Example

Nominative

Objective

Subjective

Dative

Ablative

Genitive

Locative

Instrumental

Vocative

**4.6.2.1 Tense**

Sinhala language has two tenses. 1. Non past tense 2. Past tense Both future and present tenses are represented by Non past tense. Verb conjugations differ based on tenses. We have described other characteristics under the subheading “Nouns”. The verb conjugations in Sinhala Language can be illustrated as follows. We have used the verb root “ක” (to eat) for the illustration.

Since we are focusing on Sinhala, we undertook a thorough study of Sinhala grammar [3]. As our supervisor, Professor Gihan Dias, advised, we referred several books to understand Sinhala grammar. According to the sources we referred, we divided the words in Sinhala language into 4 different categories as,

1. නාම පද
2. ක්‍රියා පද
3. උපසර්ග
4. නිපාත

During our study about Sinhala Grammar rules, we identify some basic solid grammar rules in Sinhala language which we can directly take into our project. Those are as follows.

* If the subject is singular, the verb should also be singular. If the subject is plural, then the verb should be in plural terms.
* The verb should be in the same person as it is in subject. As an example, if the subject is 1st person like මම, the verb should be in the 1st person style like යමි, කමි, දුවමි.
* The verb should be in the preferred tense of the sentence. If we consider the verb යනවා in its 1st person singular style, it should be written as ගියෙමි, යමි or යන්නෙමි respectively in past tense, present tense and future tense sentences.
* If the noun is non-living or flora, the verb should in singular style irrespective to the singularity or plurality of the noun.

**Morphology**

Sinhala is a highly morphological language. Both nouns and verbs are combined with morphemes to form different verb or noun forms according to the cases in Sinhala

Ex: ළමයා -> ළමයාට

කර -> කරමි

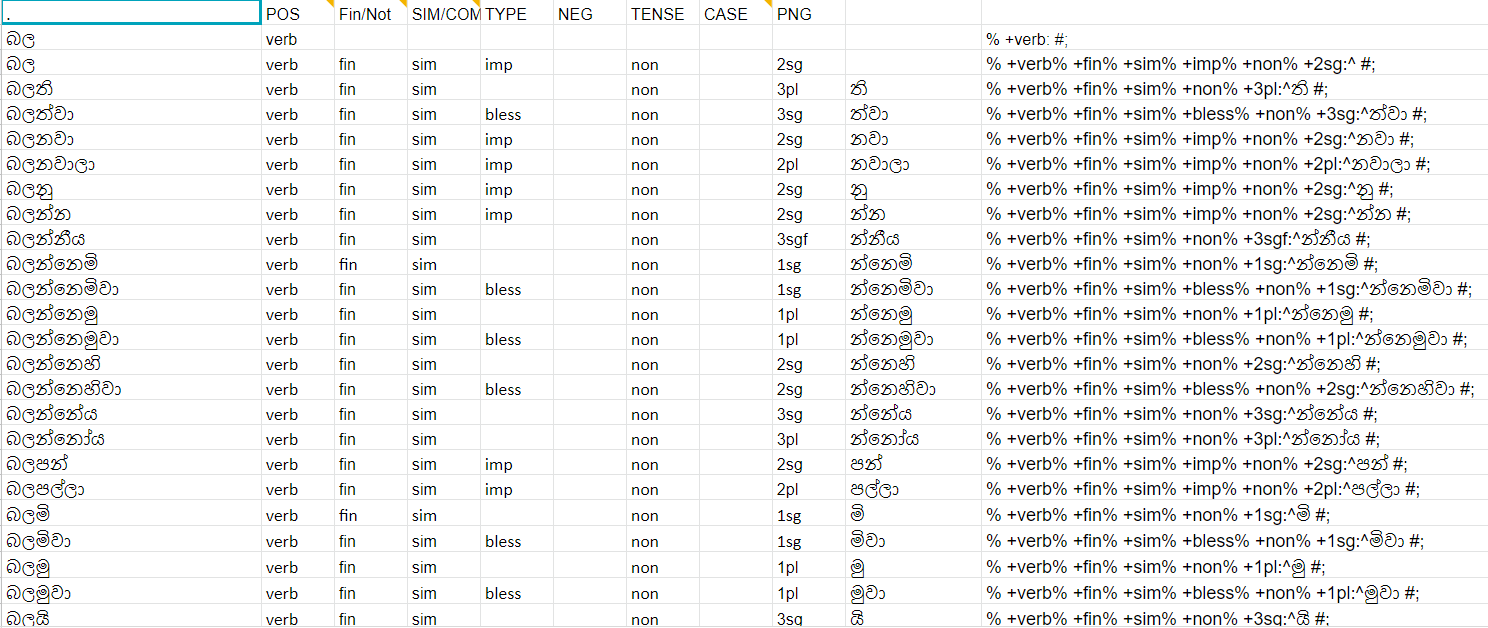
Nouns

Morphemes are added to nouns to make cases of nouns. The position of the speech depends on theses morphemes

|  |  |
| --- | --- |
| Cases | Morpheme |
| accusative | ළමයා විසි​න් |
| dative | ළමයා​ට |
| ablative | ළමයාගෙ​න් |
| genitive | ළමයා​ගෙ |
| vocative | ළමයිනි, |

Verbs

There are over 60 verb forms when combined with morphemes.



## Research on Existing Tools and Technologies

According to our research we found two approaches which people have taken to develop and parse the grammar rules in different languages. Those are the lexical functional grammar approach and the deep learning approach. Some developers have used a combination of those 2 approaches for their project.

We came across some modern language parsers in the world such as ‘Stanford Neural Network Dependency Parser’ which is currently supporting English and Chinese. This parser uses the transition-based dependency parsing where the machine learning model should be trained using an annotated corpus such as universal dependencies.

The Xerox Linguistic Environment (XLE) [1] which was developed by the Palo Alto research center (PARC) is an advanced tool to develop lexical functional grammar. This has been used to develop grammar tools to many languages as English, French, German, Urdu and Japanese.

Many low resource languages use XLE [1] for grammar development as the other machine learning based parsers need a big annotated set of data to train the model, which is not available for a low resource language. Sinhala as a low resource language faces the same problem in using a machine learning base parser to parse the grammar. So we identified XLE as the grammar parsing tool to be used to take our first steps towards the Sinhala Grammar Checker.

According to research we found that we can use finite automata for developing a morphological analyzer since we are following an approach biased to rules based. Through finite automata we can add morphemes to a root verb and develop other forms of verbs as well. If a certain form of word is given reverse engineering can be also done to generate other possible outcomes of that word.

We came across XFST FOMA and AT&T tools as finite state automata tools. Although XSFT is developed by Palo Alto team we decided to use FOMA. FOMA [17] is an alternative for XSFT. This was developed by Mans Hulden. FOMA is compatible with Xerox and PARC tools. This supports Unicode and various formats for specifying regular expressions: the Xerox/PARC format, a Perl-like format, and a mathematical format that takes advantage of the 'Mathematical Operators' Unicode block. FOMA can be directly combined with XLE.

# PROJECT WORKFLOW

## Background Study

We Studied the project report of a previous project ‘Maharawana’, done by 07th batch students in 2011. They have developed their software according to 47 simple grammar rules. They didn’t have a method to handle new words which is not included in their vocabulary. We started following the course on Grammar development by Prof. Miriam Butt. There we studied how to implement a computational grammar fragment within LFG (Lexical-Functional Grammar), worked with the grammar development platform XLE and learned how to integrate finite-state morphological analyzers into the grammar. We studied the Q&A system developed by the Palo Alto Research Center. They have used a finite state morphological analyzer in it.

In order to be eligible to use the XLE, we Applied for PARC license and obtained XLE toolkit officially. We contacted ‘Maharawana’ project members and arranged a meeting on January 31st, 2019. They acknowledged us about the approach they took, the technologies they used, the problems they faced and the achievements they had. We translated ParGram sentences to Sinhala. ParGram sentences are a set of 100 parallel sentences which has been written in many languages in the world. These sentences cover all the basic grammar rules in a language. We reviewed our translation with miss Nimasha and did the necessary adjustments. We were planning to use those sentences as a guide when developing the grammar rules.

We read the research paper, "Spell and Grammar Checking Tool for Sinhalese" by SLIIT [ref]. Their literature review helped us to get a good idea about the current technological background related to Sinhala language analysis tools. We spent more time on studying the background of the project. We finished our literature review from the knowledge we collected by reading related research papers and talking to experts in the field. We prepared our project proposal report and the presentation by including all our key findings.

## Grammar Rules Development

We spent the next 2 months (March and April) on developing the grammar rules for Sinhala using XLE. We had the opportunity to participate for the ISSALE-2019, Spring school for grammar engineering. There we learnt about the linguistics of South Asian languages and we got hands on experience with using XLE for grammar development. We started developing Sinhala grammar rules following the ParGram sentences. After some effort we understood that the order of ParGram sentences are not the best way to start developing grammar for Sinhala. So, we continued the grammar development according to our own way which is easier for us.

We planned the flow of the grammar checker tool which we are going to develop. The user can type or paste the sentences on the interface we are providing. Then we feed the sentences to the XLE. The XLE tokenizer tokenize the sentences into words. These words will be set to the morphological analyzer. The analyzer identifies and tags verbs and nouns. The unidentified words (mostly adjectives and adverbs) will be tagged using XLE tagged data set. All these tagged words will be parsed using XLE grammar rules. Then the user will be notified if there are any grammatical errors in these sentences by underlining the wrong sentences. The system auto-corrects obvious mistakes and give suggestions if there is an ambiguity about the mistake.

We collected the Sinhala schoolbooks from grade 1-5 and used them as a guideline to develop the grammar rules in XLE step by step parsing the sentences in those books from grade 1 to 5. As the first step of the grammar rule development, we parsed simple 3-word sentences in the general Sinhala word order Subject-Object-Verb. Then we developed the grammar to handle more complex sentences as mentioned below,

* Handling Adjectives and Adverbs in sentences.
* Handling Adjective Phrases and/or Prepositional Phrases in sentences.
* Handling Cases in Sinhala Language (nominative, accusative, genitive, dative).
* Handling secondary objects in the sentences

Here are some screenshots of the results generated by our web interface regarding the grammar checking of sentences.

A screenshot of a cell phone

Description automatically generatedA picture containing screenshot

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA picture containing screenshot

Description automatically generatedA picture containing screenshot

Description automatically generated

## Morphological Analyzer for FOMA

**Morphology in sinhala language based on person, gender, tense..**

Sinhala is a language rich in morphology. It gives so much information about the word. As an example, consider the word නටවන්නීය. This is made up of the root word ‘නට’ and the morpheme ‘වන්නීය’. This word can be further divided into 3 parts as bellow,

නට + ව + න්නීය

Here, the root ‘නට’ denotes that this word is regarding the act ‘dancing’. The letter ‘ව’ gives the idea that someone if forcing another to do the act. From the final section ‘න්නීය’, we can say that the act is done by a female person. The compound word with the morpheme gives a detailed idea about the context. These details are vital in checking the grammar of the relevant sentence.

Sinhala language has several types of words as Nouns, Verbs, Adjectives and Adverbs. Out of these word types sinhala verbs have the widest range of morphology. These morphologies are based on person, gender, tense and several other features. A single sinhala verb root have more than 200 morphemes where each morpheme generates a verb with a different meaning and quality. During our research, we have identified more than 80 morphemes which is used very often.

Here are the 86 verb derivatives we identified for the verb root ‘බල’,

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | POS | Fin/Not | SIM/COM | TYPE | TENSE | INSTANCE | PNG | Morpheme |
| බලති | verb | fin | sim |  | non |  | 3pl | ති |
| බලත්වා | verb | fin | sim | bless | non |  | 3sg | ත්වා |
| බලනවා | verb | fin | sim | imp | non |  | 2sg | නවා |
| බලනවාලා | verb | fin | sim | imp | non |  | 2pl | නවාලා |
| බලනු | verb | fin | sim | imp | non |  | 2sg | නු |
| බලන්න | verb | fin | sim | imp | non |  | 2sg | න්න |
| බලන්නීය | verb | fin | sim |  | non |  | 3sgf | න්නීය |
| බලන්නෙමි | verb | fin | sim |  | non |  | 1sg | න්නෙමි |
| බලන්නෙමිවා | verb | fin | sim | bless | non |  | 1sg | න්නෙමිවා |
| බලන්නෙමු | verb | fin | sim |  | non |  | 1pl | න්නෙමු |
| බලන්නෙමුවා | verb | fin | sim | bless | non |  | 1pl | න්නෙමුවා |
| බලන්නෙහි | verb | fin | sim |  | non |  | 2sg | න්නෙහි |
| බලන්නෙහිවා | verb | fin | sim | bless | non |  | 2sg | න්නෙහිවා |
| බලන්නේය | verb | fin | sim |  | non |  | 3sg | න්නේය |
| බලන්නෝය | verb | fin | sim |  | non |  | 3pl | න්නෝය |
| බලපන් | verb | fin | sim | imp | non |  | 2sg | පන් |
| බලපල්ලා | verb | fin | sim | imp | non |  | 2pl | පල්ලා |
| බලමි | verb | fin | sim |  | non |  | 1sg | මි |
| බලමිවා | verb | fin | sim | bless | non |  | 1sg | මිවා |
| බලමු | verb | fin | sim |  | non |  | 1pl | මු |
| බලමුවා | verb | fin | sim | bless | non |  | 1pl | මුවා |
| බලයි | verb | fin | sim |  | non |  | 3sg | යි |
| බලව | verb | fin | sim | imp | non |  | 2sg | ව |
| බලවති | verb | fin | sim | caus | non |  | 3pl | වති |
| බලවන්නීය | verb | fin | sim | caus | non |  | 3sgf | වන්නීය |
| බලවන්නෙමි | verb | fin | sim | caus | non |  | 1sg | වන්නෙමි |
| බලවන්නෙමු | verb | fin | sim | caus | non |  | 1pl | වන්නෙමු |
| බලවන්නෙහි | verb | fin | sim | caus | non |  | 2sg | වන්නෙහි |
| බලවන්නෙහු | verb | fin | sim | caus | non |  | 2pl | වන්නෙහු |
| බලවන්නේය | verb | fin | sim | caus | non |  | 3sg | වන්නේය |
| බලවන්නෝය | verb | fin | sim | caus | non |  | 3pl | වන්නෝය |
| බලවමි | verb | fin | sim | caus | non |  | 1sg | වමි |
| බලවමු | verb | fin | sim | caus | non |  | 1pl | වමු |
| බලවයි | verb | fin | sim | caus | non |  | 3sg | වයි |
| බලවහි | verb | fin | sim | caus | non |  | 2sg | වහි |
| බලවහු | verb | fin | sim | caus | non |  | 2pl | වහු |
| බලවු | verb | fin | sim | imp | non |  | 2pl | වු |
| බලහි | verb | fin | sim |  | non |  | 2sg | හි |
| බලහිවා | verb | fin | sim | bless | non |  | 2sg | හිවා |
| බලහු | verb | fin | sim | imp | non |  | 2pl | හු |
| බලහුවා | verb | fin | sim | caus | non |  | 2pl | හුවා |
| බලාවා | verb | fin | sim | bless | non |  | 3sg | ාවා |
| බලමින් | verb | nonf | sim |  | non | mixV |  | මින් |
| බලා | verb | nonf | sim |  | non | preV |  | ා |
| බලන්නට | verb | nonf | sim |  | non |  |  | න්නට |
| බලවන්නට | verb | nonf | sim |  | non |  |  | වන්නට |
| බලනලද | verb | nonf | sim |  | past |  |  | නලද |
| බලනුලැබූ | verb | nonf | sim |  | past |  |  | නුලැබූ |
| බලවමින් | verb | nonf | sim |  | non |  |  | වමින් |
| බලද්දී | verb | nonf | sim |  | non |  |  | ද්දී |
| බලනලදී | verb | fin | sim |  | past |  |  | නලදී |
| බැලුණු | verb | nonf | sim |  | past |  |  | ුණු |
| බැලූ | verb | nonf | sim |  | past |  |  | ූ |
| බැලුවෙමි | verb | fin | sim |  | past |  | 1sg | ුවෙමි |
| බැලුවෙමු | verb | fin | sim |  | past |  | 1pl | ුවෙමු |
| බැලුවෙහි | verb | fin | sim |  | past |  | 2sg | ුවෙහි |
| බැලුවෙහු | verb | fin | sim |  | past |  | 2pl | ුවෙහු |
| බැලුවාය | verb | fin | sim |  | past |  | 3sgf | ුවාය |
| බැලුවේය | verb | fin | sim |  | past |  | 3sg | ුවේය |
| බැලුවෝය | verb | fin | sim |  | past |  | 3pl | ුවෝය |
| බැලේ | verb | fin | sim | passive | non |  | 3sg | ේ |
| බැලෙමි | verb | fin | sim | passive | non |  | 1sg | ෙමි |
| බැලෙමු | verb | fin | sim | passive | non |  | 1pl | ෙමු |
| බැලියෙහි | verb | fin | sim | passive | non |  | 2pl | ෙහි |
| බැලෙහු | verb | fin | sim | passive | non |  | 2pl | ෙහු |
| බැලෙයි | verb | fin | sim | passive | non |  | 3sg | ෙයි |
| බැලෙති | verb | fin | sim | passive | non |  | 3pl | ෙති |
| බැලවෙයි | verb | fin | sim | passive | non |  | 3sg | වෙයි |
| බැලවෙති | verb | fin | sim | passive | non |  | 3pl | වෙති |
| බැලවේ | verb | fin | sim | passive | non |  | 3sg | වේ |
| බැලිය | verb | nonf | sim |  | non |  |  | ිය |
| බැලුවොත් | verb | nonf | sim |  | non |  |  | ුවොත් |
| බැලුවේ | verb | nonf | sim |  | past |  |  | ුවේ |
| බලන්නේ | verb | nonf | sim |  | non |  |  | න්නේ |
| බලන | verb | nonf | sim |  | non |  |  | න |
| බලනට | verb | nonf | sim |  | non |  |  | නට |
| බලවා | verb | nonf | sim |  | non | preV |  | බලවා |
| බලවන්නේ | verb | nonf | sim | caus | non |  |  | වන්නේ |
| බලනුයේ | verb | nonf | sim |  | non |  |  | නුයේ |

**Why we need a morphological analyzer to check grammar?**

In Sinhala language there are several agreements that we need to consider when generating a grammatically correct sentence. Subject - Verb agreements are one of the major agreements in Sinhala language (This has been explained above under Literature review on Sinhala language). Consider the verb ‘බලන්නීය’. This verb needs a third person, singular, animate and feminine word as the subject like ‘නංගි’ or ‘ගුරුතුමිය’. The verb ‘බලහු’ derives from the same verb root ‘බල’ as ‘බලන්නීය’ requires a second person, plural, animate word as the subject (eg: තොපි,ඔබලා). Verbs derives from some verb roots like ‘දෙ’ can have no object, only a primary object or both primary and secondary objects with in the sentence. Eg:

* මම දුන්නෙමි (no objects)
* ඔහු පොතක් දුන්නේය (only a primary object)
* ගුරුතුමිය ළමයාට පොතක් දෙයි (both primary and secondary objects)

Verbs derives from some verb roots like ‘බල’ can have no object or only a primary object in the sentence. There cannot be a secondary object for such a verb. Eg:

* මම බැලුවෙමි (no objects)
* ඔහු රුපවාහිනිය බැලුවේය (only a primary object)

These 2 sentences are grammatically correct possible Sinhala sentences but there can’t be a sentence like ‘ඔහු ඇයට රුපවාහිනිය බලයි’.

To check these kind of grammar, we need several information about the words in the sentence. To get those morphological information about each word, we need a morphological analyzer in our grammar checker tool. There are around 1000 verb roots in Sinhala language. During our research, we have identified 350+ verb roots which are used very often.

Consider the verb root ‘බල’. The 3rd person, singular, masculine and past tense verb form of this root is ‘බැලුවේය’ which can be generated by adding the morpheme ‘ුවේය’ to the root verb. Same way, we can generate the 3rd person, singular, masculine and past tense verb from of verb roots like ‘හද’, ‘සොය’, ‘පිරික්ස’ and ‘හරිගස්ස’ by adding the same morpheme, ‘ුවේය’ to the end of the verb root. The results would be හැදුවේය, සෙයුවේය, පිරික්සුවේය and හරිගස්සුවේය. But if we consider a verb root like ‘වඳී’, we have to add the morpheme ‘්දේය’ to generate the 3rd person, singular, masculine and past tense verb form ‘වැන්දේය’. We can identify that the verb roots like පලඳි, පිඹි, ලුහුබඳි and නඟි behave in the same way as the root ‘වඳී’ in forming their derivatives.

|  |  |  |
| --- | --- | --- |
| Verb Root | Morpheme | Verb |
| බල |  | බැලුවේය |
| තබ |  | තැබුවේය |
| ඉල්ල |  | ඉල්ලුවේය |
| පෙන්ව | ුවේය | පෙන්වුවේය |
| ඉක්මව |  | ඉක්මවුවේය |
| උගන්ව |  | ඉගැන්වුවේය |
| අතුල්ල |  | ඇතිල්ලුවේය |
| වඳී |  | වැන්දේය |
| පලඳි |  | පැලැන්දේය |
| බිඳි |  | බින්දේය |
| ලුහුබඳි | ්දේය | ලුහුබැන්දේය |
| හිඳි |  | හින්දේය |
| ඇහිඳි |  | ඇහින්දේය |
| නවති |  | නැවතුනේය |
| පරදි |  | පැරදුනේය |
| වරදී | ුනේය | වැරදුනේය |
| මුදාහරි |  | මුදාහැරුනේය |
| සුරකි |  | සුරැකුනේය |
| අරගන් |  | අරගත්තේය |
| අහගන් |  | අහගත්තේය |
| සමුගන් | තේය | සමුගත්තේය |
| ගොඩගන් |  | ගොඩගත්තේය |
| තියාගන් |  | තියාගත්තේය |

Here you can see that different verb roots behave in different manners. With the help of Ms. Nimasha Dilshani we divided all the verb roots into 17 Classes so that the verb roots belong to the same class have similar patterns when forming derivatives.

17 verb classes we considered,

1. අ ගණය
2. අදි ගණය
3. අරි ගණය
4. වඳී ගණය
5. උපදි ගණය
6. නවති ගණය
7. එ ගණය
8. ගන් ගණය
9. ක ගණය
10. වෙ ගණය
11. ය ගණය
12. දෙ ගණය
13. බො ගණය
14. කර ගණය
15. ලබ ගණය
16. බො ගණය
17. ඉන් ගණය

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
| අ ගණය | ඉ ගණය | | | | | එ ගණය | අනියම් ගණය | | |
| ගස | (අදි ගණය) | (වඳි ගණය ) | (අරි ගණය) | (නවති ගණය ) | (උපදි ගණය ) |  | ගන් ගණය | ක ගණය | (වෙ) |
| වස | අයැදි | අඳි | වළකි | නවති | උපදි | වැටෙ | අරගන් | ක | රොක්වෙ |
| අඹ | අනි | ඇහිඳි | සුරකි | නසි | වදි | ඈඳෙ | අහ ගන් | ගා | පහවෙ |
| තබ | ගනි | පලඳි | වඩි | පරදි | උගනි | හැපෙ | සමු ගන් | ද | බලාපොරොත්තුවෙ |
| දම | දනි | බඳි | හරි | පවති |  | දැනෙ | ගොඩගන් | ල | පසුවෙ |
| දර | පිළිපදි | මඳි | සිටි | පහදි |  | පෑදෙ | තියාගන් | නා | පාළුවෙ |
| පත | අමදි | ලුහුබඳි | වදි | වළකි |  | ගැටෙ | තියාගන් | පා | වැඩිවෙ |
| වස | ඇවිදි | වඳි | පිසි |  |  | තේරෙ | දිනා ගන් | බා | වැයවෙ |
| හද | පදි | විඳි | නසි |  |  | පෙනෙ | පිළි ගන් | වා | හෙළිවෙ |
| සද | පිරිමදි | සිඳි | හරි |  |  | වැදෙ | භාර ගන් | හා | මතක්වෙ |
| නග | පිළිපදි | නඟි | අරි |  |  | හැරෙ | හට ගන් | රඟපා | මුසුවෙ |
| නව | බදි | ඉඹි | හඟි |  |  | ලැබෙ | ගන් | සා | පසුවෙ |
| සර | මදි | සිඹි | මුදාහරි |  |  | යෙදෙ | පවත්වා ගන් |  | පාළුවෙ |
| මර | නගි | පලඳි | අඟි |  |  | මොරදෙ | හඳුනාගන් |  | පිටත්වෙ |
| අන | නැගිටි | හිඳි | අමදි |  |  | පැතිරෙ | කරගන් |  | එක්වෙ |
| අය | පනි | අනුබඳි | අහි |  |  | ඉපදෙ | පනවාගන් |  | හිරවෙ |
| අස | බසි | කල්බඳි | ඇනහිටි |  |  | වැටහෙ |  |  | පළවෙ |
| බල | නගි | පලඳි | කටි |  |  | එළඹෙ |  |  | අස්වෙ |
| කඩ | පනි | පිඹි | කාවදි |  |  | ගැලපෙ |  |  | හමුවෙ |
| කප | රකි | බඳි | කිබිසි |  |  | දික්ගැස්සෙ |  |  | වෙ |
| කව | දකි | බිඳි | කිරිවදි |  |  | නියැලෙ |  |  | සමත්වෙ |
| කස | අක්දකි | සිඹි | කිවිසි |  |  | ගැළපෙ |  |  | සිදුවෙ |
| ගය | අගි | සිඳි | කුමුදි |  |  | පැටළෙ |  |  | වෙ |
| ගර | අත්දකි |  | පරදි |  |  | හැසිරෙ |  |  | එරෙහිවෙ |
| ගස | අදි |  | පිසි |  |  | පැවරෙ |  |  | පදිංචිවෙ |

Although the verb roots belong to same class show similarities when forming derivatives, those verbs roots do not behave in the exact same manner. Let’s compare the behaviours of some verb roots in class 01 - අ ගණය.

|  |  |  |
| --- | --- | --- |
| Verb Root | Morpheme | Verb |
| ඉර |  | ඉරුවේය |
| එල්ල |  | එල්ලුවේය |
| දින |  | දිනුවේය |
| නෙළ |  | නෙළුවේය |
| පිරික්ස |  | පිරික්සුවේය |
| කඩ |  | කැඩුවේය |
| ගස |  | ගැසුවේය |
| මර |  | මැරුවේය |
| ගොඩනග |  | ගොඩනැගුවේය |
| අස්වස |  | අස්වැසුවේය |
| කොට | ුවේය | කෙටුවේය |
| දොඩ |  | දෙඩුවේය |
| සොය |  | සෙයුවේය |
| අකුළ |  | ඇකිළුවේය |
| අතුර |  | ඇතිරුවේය |
| අවුස්ස |  | ඇවිස්සුවේය |
| දුව |  | දිවුවේය |
| පුද |  | පිදුවේය |
| උදුර |  | ඉදිරුවේය |
| උසුල |  | ඉසිලුවේය |

When generating the 3rd person, singular, masculine and past tense verb form we add ‘ුවේය’ suffix at the end of each verb root which is mentioned above. But we can see that the root part of the verb has faced a slight modification and that modification is in different patterns in different cases. Consider the first 5 verb roots, ඉර, එල්ල, දින, නෙළ and පිරික්ස. When concatenating with morphemes to form the respective verbs, the root part of the verb has not faced any change.

**ඉර+**ුවේය, **එල්ල+**ුවේය, **දින+**ුවේය, **නෙළ+**ුවේය, **පිරික්ස+**ුවේය

But when we consider the next 5 verb roots, we can clearly see that the root part of the derived verb has faced a slight change.

|  |  |
| --- | --- |
| Original Verb Root | Modified Verb Root |
|  |  |
| කඩ | කැඩ |
|  |  |
| ගස | ගැස |
|  |  |
| මර | මැර |
|  |  |
| ගොඩනග | ගොඩනැග |
|  |  |
| අස්වස | අස්වැස |

This way, we can identify a set of subgroups within each main group.

A screenshot of a cell phone

Description automatically generated

*Figure 4: Different Subclasses of Class-01*

As we described earlier, we are using FOMA as the technology to develop our verb morphology analyzer. In FOMA we have to provide all the verb roots and the set of morphemes related to each group of verb roots in a lexicon data file. For each lexicon file, we should apply a unique set of regular expressions (re-write rules) to generate the verb forms correctly. A lexicon file related to several subclasses of class - A (අ ගණය ) is shown below.

**LEXICON ClassA5**

සපුර CA5Vinf;

හඳුන CA5Vinf;

හඳුන්ව CA5Vinf;

පමුණුව CA5Vinf;

හදාර CA5Vinf;

……….. …………..

……….. …………..

අහුර CA5Vinf;

අවුස්ස CA5Vinf;

**LEXICON CA5Vinf**

% +verb:^ #;

% +verb% +fin% +sim% +imp% +non% +2sg:^ #;

% +verb% +fin% +sim% +non% +3pl:^ති #;

% +verb% +fin% +sim% +bless% +non% +3sg:^ත්වා #;

% +verb% +fin% +sim% +imp% +non% +2sg:^නවා #;

% +verb% +fin% +sim% +imp% +non% +2pl:^නවාලා #;

…………………………………………………………………….

…………………………………………………………………….

% +verb% +nonf% +sim% +non:^නට #;

% +verb% +nonf% +sim% +non% +preV:^බලවා #;

% +verb% +nonf% +sim% +caus% +non:^වන්නේ #;

% +verb% +nonf% +sim% +non:^නුයේ #;

The easiest way to manage sub classes is creating a lexicon file for each subclass separately. But then we will have to create more than 200 lexicon files which is not practical. So we have combined several subclasses into one lexicon file to minimize the number of total files created.

There should be a foma file to call these lexicon files and generate the final verb. In this foma file, we can define regular expressions to modify the verb root as needed (See Table xx above).

These regular expressions are described below.

**define Ba5** [ ු ණ ු | ූ | ු ව ෙ ම ි | ු ව ෙ ම ු | ු ව ෙ හ ි | ු ව ෙ හ ු | ු ව ා ය | ු ව ේ ය | ු ව ෝ ය | ව ේ | ෙ ම ි | ෙ ම ු | ෙ ය ි | ෙ හ ු | ෙ ත ි | ෙ හ ි | ව ෙ ය ි | ව ෙ ත ි | ේ | ි ය | ු ව ො ත ් | ු ව ේ ];

**read lexc a5.lexc**

**define Lexicona5;**

define PathuraClass1 [..] -> ැ || .#. A \_ A ?\* "^" Ba5 ;

define PathuraClass2 ු -> ි || .#. A ැ A \_ ?\* "^" Ba5 ;

define AkulaClass1 අ -> ඇ || .#. \_ ?\* "^" Ba5 ;

define UduraClass1 උ -> ඉ || .#. \_ ?\* "^" Ba5 ;

define AkulaClass2 ු -> ි || .#. [ඇ | ඉ] A \_ ?\* "^" Ba5 ;

define UganwaClass2 [..] -> ැ || .#. ඉ A \_ A ?\* "^" Ba5 ;

define KalaththaClass2 [..] -> ැ || .#. A ැ A \_ A ් A "^" Ba5 ;

define HadaaraClass2 ා -> ෑ || .#. A ැ A \_ A "^" Ba5 ;

define PupuraClass1 ු -> ි || .#. A \_ A ?\* "^" Ba5 ;

define PupuraClass2 ු -> ි || .#. A ි A \_ A ?\* "^" Ba5 ;

define Cleanup "^" -> 0;

**define Classa5Grammar** Lexicona5 .o.

PathuraClass1 .o.

PathuraClass2 .o.

AkulaClass1 .o.

UduraClass1 .o.

AkulaClass2 .o.

UganwaClass2 .o.

KalaththaClass2 .o.

HadaaraClass2 .o.

PupuraClass1 .o.

PupuraClass2 .o.

Cleanup;

Here you can see that we have defined an array called Ba5 with some morphemes. These are the set of morphemes which the rewrite rules should be applied. If we take the verb root ‘සපුර’, the creation of verb forms like සපුරමි, සපුරති, සපුරපන්, සපුරා do not need any alteration in their verb root. But the creation of verb forms like සැපිරුවාය, සැපිරුවෙමි, සැපිරිණි, සැපිරෙන need to alter the verb root ‘සපුර’ into ‘සැපිර’. So that the first thing we need to do is identify the sets of morphemes which need alterations in the verb root and categorize them under separate arrays so that we can apply the rewrite rules separately to each set of morphemes.

**Challenges in using FOMA**

As computational grammar is not a field which is studied by a huge sector of people and FOMA is not a widely spread technology, the online community is limited. So we had to carry on our development with a limited community support. FOMA is a simple compiler and a library developed using the C language. There are no debugging or other sophisticated supports in FOMA. Some functionalities we needed to alter the verb roots are not available in FOMA, so we had to use alternative approaches. As an example, consider the verb root ‘අදි’. To generate the past tense verb forms like ඇද්දේය, ඇද්දෝය, ඇද්ද and some commanding verb forms like ඇදපන්, ඇදපල්ල need to alter the root ‘අදි’ into ‘ඇද්ද’. Same way the below alterations should have to be done to the other verb roots in the same class.

|  |  |
| --- | --- |
| Original Verb Root | Modified Verb Root |
|  |  |
| අදි | ඇද්ද |
|  |  |
| ගනි | ගැන්න |
|  |  |
| නැගිටි | නැගිට්ට |
|  |  |
| ඉසි | ඉස්ස |
|  |  |
| අස්වදි | අස්වැද්ද |

To do this alteration, we need to double the last letter of the verb root. But there is no such functionality in FOMA. So we had to manually replace the last letter accordingly using the following rewrite rule,

Define CERule1 ත -> ත ් ත || \_ "^" [BE1 |EE1 ];

Define CERule2 ද -> ද ් ද || \_ "^" [BE1 |EE1 ];

Define CERule3 න -> න ් න || \_ "^" [BE1 |EE1 ];

Our current verb collection has 354 verb roots and each verb root has around 70 derived verbs. All together our verb morphological analyzer can currently handle more than 24000 different verbs. We developed a simple web interface by connecting our verb morphological analyzer as the backend. There a user can enter any verb and get the tag set related to that verb as the output.

A screenshot of a cell phone

Description automatically generated

*Figure 5: Morphological Analyzer UI*

Other than this stand alone web interface, we have developed a simple API to retrieve these data in json format. This API can be used by developers in their projects.

# FUTURE WORK

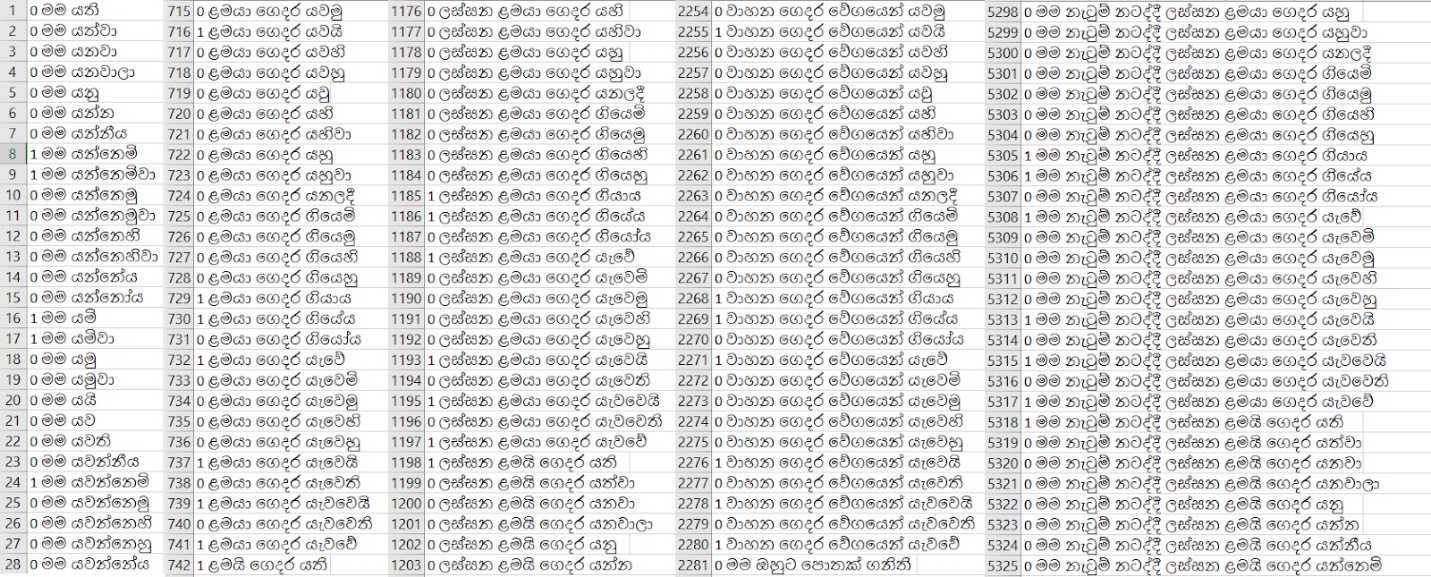
This project requires an immense amount of research on Sinhala language and technologies related to linguistic development. Our research on Sinhala language will continue throughout the project. We need to modify the remaining areas of Sinhala grammar using XLE. They are handling the free word order and handling passive voice sentences. Morphological analyzer will be completed with the help of Sinhala Spell checker team (the spell checker team provides the noun morphology analyzing part). Next month (December) will be spent for documentation including research papers.

# Testing and Validation

We used Syntax based coverage methodology for testing.

**Production based coverage:** testing requirement contains each production P in grammar G

We developed a test dataset of 16920 labeled sentences. The labeling was done using python scripts and manual effort.



A screenshot of a cell phone

Description automatically generated

# CONCLUSION

Sinhala Grammar checker is a tool which is required by the majority of the Sri Lankans who use Sinhala language for documentation. The objective is to deliver a Sinhala Grammar Checker as a plug-in for OpenOffice and LibreOffice packages. The Grammar checker will be developed using a hybrid approach of Lexical Functional grammar and Deep learning. The project is planned to be completed within one year.

# REFERENCES

[1] Butt, M. and King, T. (2002). Urdu and the Parallel Grammar Project. In: *COLING '02 Proceedings of the 3rd workshop on Asian language resources and international standardization*. Association for Computational Linguistics.

[2] Ling.uni-konstanz.de. (2019). *XLE Project*. [online] Available at: http://ling.uni-konstanz.de/pages/xle/ [Accessed 8 Feb. 2019].

[3] Karunathilaka, W. (2015). *Sinhala Bhasha Wiyakaranaya*.

[4] En.wikipedia.org. (2019). *Lexical functional grammar*. [online] Available at: https://en.wikipedia.org/wiki/Lexical\_functional\_grammar [Accessed 8 Feb. 2019].

|  |  |
| --- | --- |
| [5] | Z. Yuan, T. Briscoe and M. Felice, "Candidate re-ranking for SMT-based grammatical error correction," The ALTA Institute Computer Laboratory, University of Cambridge, Cambridge, 2016. |
| [6] | G. Melis, C. Dyer and P. Blunsom, "On the state of the art of evaluation in Neural Languge Models," University of Oxford, Oxford, 2018. |
| [7] | B. Hettige and A. Karunananda, "A Morphological Analyzer to Enable English to Sinhala Machine Translation," in *2006 International Conference on Information and Automation*, Colombo, 2006. |
| [8] | B. Hettige and A. Karunananda, "A Parser for Sinhala Language - First Step Towards English to Sinhala Machine Translation," in *First International Conference on Industrial and Information Systems*, Peradeniya, Sri Lanka, 2006. |
| [9] | D. De Silva and A. Alahakoon, "Sinhala to English Language Translator," in *4th International Conference on Information and Automation for Sustainability*, Colombo, 2008. |
| [10] | T. N. E. Fernando and A. R. Werasinghe, "A Morphological Parser for Sinhala Verbs," in *Advances in ICT for emerging Regions*, Colombo, 2013. |
| [11] | M. Jayasooriya and A. R. Weerasinghe, "Learning a stochastic part of speech tagger for sinhala," in *International Conference on Advances in ICT for Emerging Regions (ICTer)*, Colombo, 2013. |
| [12 | A.Z. Syed, M. Aslam and A. M. Martinez-Enri, "Lexicon Based Sentiment Analysis of Urdu Text Using SentiUnits," in *Mexican International Conference on Artificial Intelligence*, Mexican City, 2010. |
| [13] | L. A. Abeyrathne and S. Edirisinghe, "Spell and Grammar Checking Tool for Sinhalese," Sri Lanka Institute of Information Technology, Malabe, Colombo, 2018. |
|  |  |
| [14] | C. Leacock and M. Chodorow, "Automated Grammatical Error Detection for Language Learners," Microsoft Research, 2010. |
| [15] | N. Nagata, "An Effective Application of Natural Language Processing in Second Language Instruction," *CALICO Journal,* vol. 13, no. 1, pp. 47-67, 1995. |
| [16] | A. Kumar and S. B. Nair, "An Artificial Immune System Based Approach for English Grammar  Checking," in *International Conference on Artificial Immune Systems*, Santos, Brazil, 2007. |
| [17] | Hulden M. “Foma: a finite-state compiler and library” in EACL '09 Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics, Athens Greece, 2009. |