Efficient Rule-Based Grammar Checker with Word Sequencing

Thesis submitted in partial fulfillment of the requirements for the

award of degree of

Master of Engineering

In

Computer Science and Engineering

Submitted By

Krishna Kumar Giri

(801532026)

Under the supervision of:

Mr. Rajkumar Tekchandani

Assistant Professor

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

THAPAR UNIVERSITY
PATIALA – 147004

JULY 2017

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, "*Efficient Rule-Based Grammar Checker with Word Sequencing*", in partial fulfilment of the requirements for the award of degree of Master of Engineering in *Computer Science and Engineering* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of Mr. Rajkumar Tekchandani and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

(Krishna Kumar Giri)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(Mr. Rajkumar Tekchandani)
Assistant Professor
Computer Science and Engineering
Department

ACKNOWLEDGEMENT

The successful completion of any task would be incomplete without acknowledging the people who made it possible and whose constant guidance and encouragement secured the success.

First of all I wish to acknowledge the benevolence of omnipotent God who gave me strength and courage to overcome all obstacles and showed me the silver lining in the dark clouds. With the profound sense of gratitude and heartiest regard, I express my sincere feelings of indebtedness to my guide **Mr. Rajkumar Tekchandani**, Assistant Professor, Computer Science and Engineering Department, Thapar University for his positive attitude, constant encouragement, keen interest, invaluable cooperation, generous attitude and above all his blessings. He has been a source of inspiration for me, I am grateful to **Dr. Maninder Singh**, Head of Department and **Dr. Ashutosh Mishra**, P.G. Coordinator, Computer Science and Engineering Department, Thapar University for the motivation and inspiration for the completion of this thesis. I will be failing in my duty if I do not express my gratitude to **Dr. S. S. Bhatia**, Senior Professor and Dean of Academics Affairs in the University for making provisions of infrastructure such as library facilities, computer labs equipped with internet facility, immensely useful for the learners to equip themselves with latest in the field.

Last but not the least I would like to express my heartfelt thanks to my parents and my friends who with their thought provoking views, veracity and whole hearted co-operation helped me in doing this thesis.

Krishna Kumar Giri
(801532026)

ABSTRACT

A grammar checker is a program which is used to correct the grammar of a sentence or a paragraph. These days, informal writing (communication on social websites and chat applications) has gained a lot of popularity. In informal writing there is no requirement of grammar to be correct. However when we come to formal writing such as research papers, emails, patents etc. then it is required that, English written in these formal writings should be free of grammar errors. Moreover, error free text explains the idea or thought more clearly.

In this thesis we present a rule-based grammar checker, which is capable of correcting the word order of a sentence in English language. Our tool first identifies the tense of the input sentence. For every tense we have different categories of sentences. Then based on the tense, it further categorizes the sentence into the category it belongs, and depending on the category of the sentence, corresponding function is called which corrects the sentence. The correction is done with the help of certain rules that we have defined for every category of sentence. The rules are based on the word order of the sentence. Our tool also incorporates the work of existing grammar checkers. This work further checks the grammar of the sentence. If the grammar of sentence is incorrect, then it is corrected after correcting the word order of the sentence.

For evaluating the performance of our tool, we have compared it with already existing nine grammar checkers. Results show that the accuracy of our grammar checker is better than these grammar checkers. Moreover, our tool is able to correct the incorrect order of the sentence which in not done by the previous researchers.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1 INTRODUCTION

A computer program that is created to inspect the correctness of a sentence, a paragraph, or a text file etc. is a grammar checker. The implementation of grammar checkers is mostly done as a feature or a part of some big application or program like Microsoft Word. Also there are stand-alone online website and applications providing the grammar correcting facility.

1.1 History of grammar checkers

Earlier, the grammar checkers used to be the programs which were equipped with the feature of providing style checking and punctuation inconsistencies, and not all the possible grammatical errors. “Writer’s Workbench” was the first grammar checker and it was incorporate in UNIX systems in 1970s. The Writer’s Workbench package comprised of various tools for checking different writing problems. The output of this grammar checker was a list of potential suggestions for improving the text.

The earliest version of style checker and diction was “Grammatik”. It was developed for personal computers, and was announced by “Aspen Software” of Albuquerque. Initially it was available for “Radio Shack – TRS - 80”, and later was accessible to IBM PC and CP/M. “Reference Software” of San Francisco, CA acquired Grammatik. Grammatik developed continuously, and became a genuine grammar checker for identifying writing errors more than simple style checking.

There were many programs for word processing in 1992, but “Microsoft Word” and “Word Perfect” were the best amongst all in the market. In year 1992, Microsoft chose to include grammar checking as a new feature to their package and licensed “CorrectText”. “CorrectText” is a grammar was also a grammar checker that was not into the market yet. In answer to this act of Microsoft, “Reference Software” was acquired by the WordPerfect.

1.2 Widely Used Ways to Implement a Grammar Checker

Rule-based checking, statistics-based checking and hybrid checking are the three widely used ways to implement a grammar checker [1]. Rule-based checking has set of rules developed manually, match against the text. It has advantages such as sentence does not need to be complete to be checked, rules can be built incrementally, suggest error message with helpful comments, easy configuration and so on. However, it's very difficult to understand and include 2 all correct grammatical rules which can be used to check sentences, especially for complex sentences. On the other hand, in the statistic-based checking, Part Of Speech (POS)-annotated corpus is used to build the grammar rules automatically by identifying list of POS tag sequences and then those common sequences that occur often can be considered correct and the uncommon ones incorrect. However, it's very difficult to understand error message suggested by this checking system as there is no specific error message [2]. In hybrid grammar checking, the good qualities of the rule-based and statistical approach can be combined to develop the grammar checker.

1.3 Fundamental concepts of Grammar Checking

1.3.1 Tokenization

When input is given to natural language processing system, it may occur or be given as a paragraph or paragraphs. This input text needs tokenization process, i.e. input text to an individual occurrence of a linguistic unit, for further processing [1]. The tokenization processes may be splitting the input text to sentences and also to words if necessary. In the case of grammar checker, input texts should be broken into sentences and then sentences to words. The output words can be tagged and analyzed separately in the morphological analyzer. Both the training and test set should pass the tokenization process while using statistical grammar checker approach. The rule-based grammar checker also needs the tokenization on the input text to be checked. Like the statistical grammar checker, the input text should be split into sentences and then to words in order to be tagged and analyzed.

1.3.2 Parts Of Speech Tagging

Part-of-speech tagging describes the assignment of the appropriate word class and morpho syntactic features to each token in a text. The process of tagging is performed by a tagger [3]. The most common POS include noun, pronoun, verb, adjective, adverb, preposition, 18 conjunction, interjection and the like [4]. Before naming the word with its POS tag, the POS tagger should look at what the word is doing in a specific sentence because some words can be used in several different ways.

POS tagger or POS tagging is one of the important components in most of the grammar checkers developed in pervious researches [6]. Learning about the parts of speech helps to understand grammar mistakes that can be made and figure out how to correct them. It's also very advantageous in grammar checking as it's very difficult to include each word of the language in the manually constructed rules. POS tagger is also used in statistical approach as it is almost impossible to find each word in an input text in the corpora. Therefore, the words in the sentence should be assigned to their part of speech tags and check the correctness of the sentence based on the assigned tag in the corpora. All these factors make POS tagging an important component for grammar checker development.

1.3.3 Corpora

Oxford dictionary define corpus as "a body or collection of linguistic data which either be written or spoken text in machine readable form". The data of the corpus are typically digital, i.e. it is saved on computers and it's machine-readable. Corpus has components [3].

Components are:-

- The text data itself, this is the text in natural language
- Possibly Meta data which describe the text data,
- And linguistic annotations related to the text data. This may be a POS tag, morphological analysis of words and linguistics information about the letters, word, 17 sentences and the like.

Many languages are under-resourced especially languages in developing countries like Ethiopia [8]. English language has also a shortage of sample annotated corpus. Due to that different English researches suffer and spend much time in collecting the available corpora and analyzing to make it suitable to their own research use [7]. Statistical grammar checker requires grammatically correct texts to training the system. The main issue related to statistical approach is finding error free corpus to train the system. The training and test data set should also be annotated i.e. words in corpus needs to be POS tagged and also morphologically analyzed. A rule-based grammar checker performs the tagging and analysis as a pre-process on the text to be checked.

Walta information center corpus is an Arhamic news corpus. It contains about 8067 sentences which are taken from walta information center English news. Single news in the corpus contains title and content. The corpus has another copy which is manually POS tagged with part of speech. Corpora, including the manually tagged, are available on the web for download.

1.4 Thesis Organization

- Chapter 1: This chapter discusses about, what a grammar checker is? And the history of grammar checkers.
- Chapter 2: This chapter discusses about the previous work done by other researchers in this field.
- Chapter 3: This chapter discusses about research statement, research gaps, research objectives and research methodologies of our work
- Chapter 4: This chapter discusses about our proposed work, the flow of our system and algorithms that we have designed.
- Chapter 5: This section discusses about the result of our proposed work and comparison of our work with previous work.
- Chapter 6: This section discusses about the whole work of our thesis in short and also discusses about future work.

Chapter 2 LITERATURE SURVEY

Basically there are three ways in which researchers have implemented grammar checker till date. All these three ways are explained below.

2.1Grammar Checking Approaches

2.1.1 Syntax Based Checker

As described in [13]. In this approach, a text is completely parsed, i.e Each sentence is analyzed and assigned a true structure. The sentence is said to be correct if the parsing succeeds and incorrect if parsing does not succeed.

There is an advantage of syntax-based approach i.e grammar checking in syntax- based approach is always complete if its grammar is complete that means any incorrect sentence will be detected, no matter how unknown the error is. Unfortunately the grammar checker will only be able to detect that there is some error, but it will not be able to suggest that what exactly the error is.

There is also a major problem of syntax based approach, it requires a complete grammar that covers every type of text needed to be checked. There are many grammar checkers but there is no robust grammar checker or a parser available today. Usually parsers give more than one than result even for correct sentences because they suffer from natural language ambiguity.

2.1.2 Statistics Based Checker– A statistical grammar checker is one of the approaches used for developing a grammar checker system. The approach assumes that a text can be corrected using a large amount text which is a collection of grammatically correct sentences [3]. In this method, the rules are generated automatically from corpus and large amount of data is required to train the system. The corpus may be built from articles, journals and other magazines. It is difficult for the users to interpret the system judgment unless the developer accesses the corpus [2]. There are two different ways of implementing statistical grammar checker. The first one is when an input text to be corrected, directly compared with the corpus. When this technique is used to check the

grammar, the corpus needs to be visited directly in every text check. The other one, generating a grammar rule from the corpus and use those rules to check the input text. This one requires updating the rules if corpus is modified or some data is added to the corpus [3]. N-gram Based Statistical Grammar Checker for Bangla [6], language independent statistical grammar checker [3] and statistical grammar checker for English [9] are some of researches done using this approach.

2.1.3 Rule Based Checker

A rule-based grammar checker approach is the most common method of grammar checking [3]. It works in a way that the input text is checked against the rules which developed manually unlike the statistical method [10]. Manual work load is one of the disadvantages of this approach. However, the rules are easy to configure and can be managed individually [2]. These rules are easy to add, remove or modify. Rules can be written by a person who has no programming experience, like linguistic professional [10]. Rule-based systems can offer a detailed error message even stating or explaining the grammar rule. It can also be developed incrementally, staring with one rule and extending the system adding rule one by one. Input sentences can also be checked on fly i.e. the sentence doesn't need to be complete to be checked, it can be checked while writing the sentence. The other thing, which makes this approach so powerful, is it can cover almost all feature of a language [3]. There are a number of grammar checker researches using rule-based approach. These include a rule-based Afan Oromo Grammar Checker [1], style and grammar checker for English [2], dependency-based rule for grammar checking [10] and Punjabi Grammar Checker [5].

2.1.4 Hybrid Grammar Checker

There are also systems that use a hybrid system of both statistical and rule-based approaches. This helps the system to achieve high efficiency and robustness [11]. "Granska", one of the Swedish grammar checkers, is developed using this approach by Rickard Domeij, Ola Knutsson, Johan Carlberger and Viggo Kann [11]. The other hybrid structure is COGrOO. It's a Portuguese grammar checker based on CETENFOLHA a Brazilian Portuguese morphosyntactic annotated Corpus [12].

2.2 Common Grammar Errors

Grammar rules and regulation are required for most of the languages, and same is true for English. If one knows rules of the language, it is very easy for others to understand what he/she writes. However, knowing all the rules may be very difficult for the user while writing specially for the non-natives ones. Therefore, people make different mistakes when they write English texts. Most of the grammar errors groups discussed in this section are common with other languages. However, this section tries to illustrate the errors in English texts. Grammar error groups are identified with the help of linguistic experts.

2.2.1 Subject-Verb Disagreement

The subject and the verb in a sentence must agree in number and person. The disagreement in all the cases creates misunderstanding or confusion for the reader of the sentence. To show each of the cases with example,

1. There are ten apple on the desk.

2. Ram go to garden every day.

In the first sentence there is subject verb-disagreement in number. The correct sentence will be “There are ten apples on the desk”. And for the second one the correct output will be “Ram goes to garden every day”

2.2.2 Incorrect Word Order

This grammar rule, the word order, has different form in different language. This word order includes the general structure of the sentence, the modifier and modified word order (Adjective and noun or adverb and verb)and the like. English has an SVO sentence structure. Depending on the natural language used, the words in a sentence should be in their correct order.

Example:

Ram every day to garden goes.

The correct sentence for the above sentence is “Ram goes to garden every day”.

2.2.3 Adjective and Noun Agreement

Modifiers are words or phrases that are used in a sentence to elaborate something. Modifiers can be Adjectives which modify noun or adverbs that modify verbs. English Adjectives appear before nouns that they modify. English Adjective should agree with the noun it modifies in number and gender. Adjective is used in a sentence to give a clear picture of the noun that it modifies. It is simply a word that tells more about the noun. English language has adjectives to modify nouns. These 14 adjectives may mark number (person and gender) and gender (feminine and masculine) of the noun they modify. If an adjective marks the number and gender of the noun, the marker should agree with the number and gender of the noun.

2.2.4 Adverb and Verb Agreement

Another type of describing word or modifier is an adverb. It defined as a category which is to a verb what an adjective is to a noun. English Adverb usually modifies the first verb that comes next to it. Like the other languages, adverbs also exist in English language. The English adverbs are few in number and can be found in primitive (as separate word that appears by themselves) and compound form (as combination of two words appearing as a word or detached words).As in many languages English adverbs are classified into subclasses such as adverbs of time, place, circumstance etc. The time adverbs describe the time at which an event takes place. These adverbs may show a specific time at which a given action takes place or its duration. English verbs indicate the time at which action takes place in relation with the adverbs.

2.3 Previous Research work in three paradigms

The previous research work can be summarized into three categories [16], machine translation, natural language generation and parsing.

2.3.1 Parsing

In parsing a complete syntactic examination of that sentence is done. And process of parsing is used to detect errors and mention corrections. There are many parsing techniques. Let us look at two most common techniques, one that relaxes rules of the grammar and one that clearly models mistakes in the grammar.

2.3.1.1 Constraint Relaxation

A way to handle a wide range of mistakes is by relaxing constraints in an unification framework [14]. Grammar errors like subject-verb agreement, are continuously until the sentence is able to be parsed. In [14] order of the composure is decided by a concept of loss of information.

An important characteristic of this approach is to parse both grammatically correct and grammatically incorrect sentence. When a grammatically incorrect stimulates the relaxation of a constraint, it becomes easy to recognize the error that is based on this constraint. There is a disadvantage of this approach i.e it there an extra or a missing word in a sentence it will not be able to parse it correctly. Besides the constraints should be pre-defined, restricting the number of mistakes that can be handled.

2.3.1.2 Mal-Rules

Another way to apply error production rules, to increase the context-free grammar, to refer to texts with errors, is also known as mal-rules. Each such rule correlates to a category of grammatical errors.

The main benefit of mal-rules is that they generate the reaction very easily but there is a disadvantage that they (mal-rules) need to pre-defined, gradually we need to handle more and more types of errors. So grammar becomes complicated. There are some systems that use the mal-rules mentioned below.

- ICICLE (Intersctive Computer Identification and Correction of Language Errors) was designed for the American sign language singers to learn English. From writing samples mal-rules were obtained. The system offers comprehensive

coverage in English rather than a proposal for improvement, it provides the reaction, such as "This phrase does not have a clutter in front of it". After this he hopes that the student will modify his writing accordingly and will re-submit it to the system

- The system focuses on performance errors in [18], it also offers error detection and correction. The system initially tries to parse with a usual CFG. If parse is not possible, it then proceeds to the error-production grammar.

  Perhaps partly to limit computational complexity, only one type of operation (insertion, deletion, substitution) can be applied to a sentence. A sentence like But not one of them is capable [of → to] [dealing → deal] with robustness as a whole, which requires two substitutions, cannot be properly treated.
- Mal-rules in Arboretum [19] are used with the LKB parser [20] in conjunction. an aligned generation strategy is significant novelty to ensure that the input sentence is as close as possible to the corrected sentence. In a subset of the English corpus [21] of Learners of Japan, an evaluation was carried on 221 items, 80.5% were correctly parsed out of these items.
- In one of the earlier applications of this prototype, the combination of graded grammar is used "It is a collection of estimating rules that are associated with a group of" rules ", which include anticipated errors, which What do we call "mal-rules".

2.3.2 Natural Language Generation

In section 2.3.2.2, carefully hand-crafted constraints and mal-rules usually give a high precision. Moreover, the grammatical errors must be anticipated in advance to be covered; typically, the set of errors is compiled using anecdotal observations from domain experts.

To remove the requirement of anticipating all the errors, a class of grammar checkers acquires methods based on Natural Language Generation (NLG) methods. NLG techniques may in be divided into two different shores. The system recognizes the important content words (“keywords”, or “concepts”), and tries to rebuild the sentence. This method is suitable for sentences that are comparatively errorful and need major repair. It typically reviews only those words in the sentence that belong to a certain part-

Chapter 3 RESEARCH PROBLEM

3.1 Problem Statement

A grammar checker is a program which is used to correct the grammar of a sentence or a paragraph. These days informal writing (communication on social websites and chat applications) has gained a lot of popularity. In informal writing there is no requirement of grammar to be correct. However when we come to formal writing such as research papers, emails, patents etc. then it is required that, English written in these formal writings should be free of grammar errors. Moreover, error free text explains the idea or thought more clearly.

3.2 Research Gaps

There are numerous grammar checkers available in the market such as grammarly, languagetool, Ginger, White Smoke, and Microsoft Word etc. These grammar checkers work on the grammar of the input text. We were analyzing these tools, and observed an issue based on grammar, on which none of these tools have worked on. The issue is that the correction of grammar of a sentence is not done according to the tense rules, i.e. the word order of a sentence is not checked by any grammar checker. We examined different types of sentences on each of the above mentioned tools, and all of them had this same issue. An example of this problem is shown below:

Example:

Incorrect sentence - Shyam playing is.

Correct sentence- Shyam is Playing.

In the above example, “is” in the incorrect sentence and should be placed before “sleeping”. But all the available grammar checkers are unable to catch this error.

3.3 Research Objectives

The following research objectives have been established, in the light of research gaps mentioned above,.

- To identify whether a sentence is a passive sentence or not. So that we can differentiate between an active and a passive sentence.
- To identify the tense of a sentence. Identifying the tense of an active sentence is a different approach than identifying the tense of a passive sentence.
- To differentiate types of sentences of each tense. Each tense consists of different types of sentences, thus every sentence needs to be identified individually.
- To correct the word order of the sentence.
- To merge the work done by different researchers till date and work proposed by us.

3.4 Research Methodology

We took a list of sentences and tested them for grammatical errors, on grammarly, languagetool, Ginger, White Smoke, and Microsoft Word. All these tools were unable to detect the error based on tense rule.

A grammar checker that corrects only the sentence ordering is not sufficient. So we have designed a grammar checker, which solves the above mentioned problem, an also enhances the work done by the existing grammar checkers. For this purpose, we need have a thorough knowledge of English grammar and tense.

In our methodology, we first take a sentence and detect its tense. To make sentences based on different tenses, we have taken various types of sentences and analyzed them. Using this analysis, we created rules for every type of sentence. Based on these rules, we determined the tense of the input new input sentences. After detecting their tense, we

positioned the elements of the sentence (subject, object, verb, adverb, adjective, preposition, etc.) according to the rules formed.

Incorrect input – The has guests arrived.

First correction – The guests has arrived.

Final correction – The guests have arrived.

In the above example, we have corrected the order of the sentence in first correction, the sentence is of present perfect tense. According to the rule of present perfect tense, verb should follow the helping verb. Based on this rule, we have corrected the grammar of the sentence. Guests is plural so with plural “have” should be used.

Chapter 4 PROPOSED WORK

4.1 Flow Chart Description

After examining the tools that are available for grammar checking, and having noted their limitations, we have designed a new grammar checker that corrects the errors based on the order of words in the sentence. Our approach is tense based. It first identifies the type of tense of the input sentence, and then corrects the grammar on the basis of rules we have made. The flow - chart in figure __ explains our approach.

Flow – chart description:

Step 1: Take input from the user in the form of a sentence.

Step 2: In this step, work of previous researches is implemented. We have combined the previous research work and our work to produce better results.

Step 3: In this step, our algorithm detects the tense of a sentence. In order to correct grammar of a sentence, every tense needs a different approach. For this purpose we have first identified the grammar of the sentence.

Step 4: Each tense is further categorized into simple tense, continuous tense, perfect tense, and perfect continuous tense. So we have collected different types of sentences in each category and made a rule for it. This is done to apply the appropriate correction rule to appropriate sentence.

Step 5: After identifying the type of rule for the sentence, a function associated with the sentence is called. As every sentence is distinct, a different rule is required to handle it. So we have created different function for different sentences.

Step 6: If the input sentence is correct according to the rules, then go to step 8, otherwise go to step 7.

Step 7: In this step, our algorithm corrects the sentence following the rules. We have arranged the words in the sentence according to noun, verb, preposition, article etc.

Step 8: Corrected output is displayed to the user.

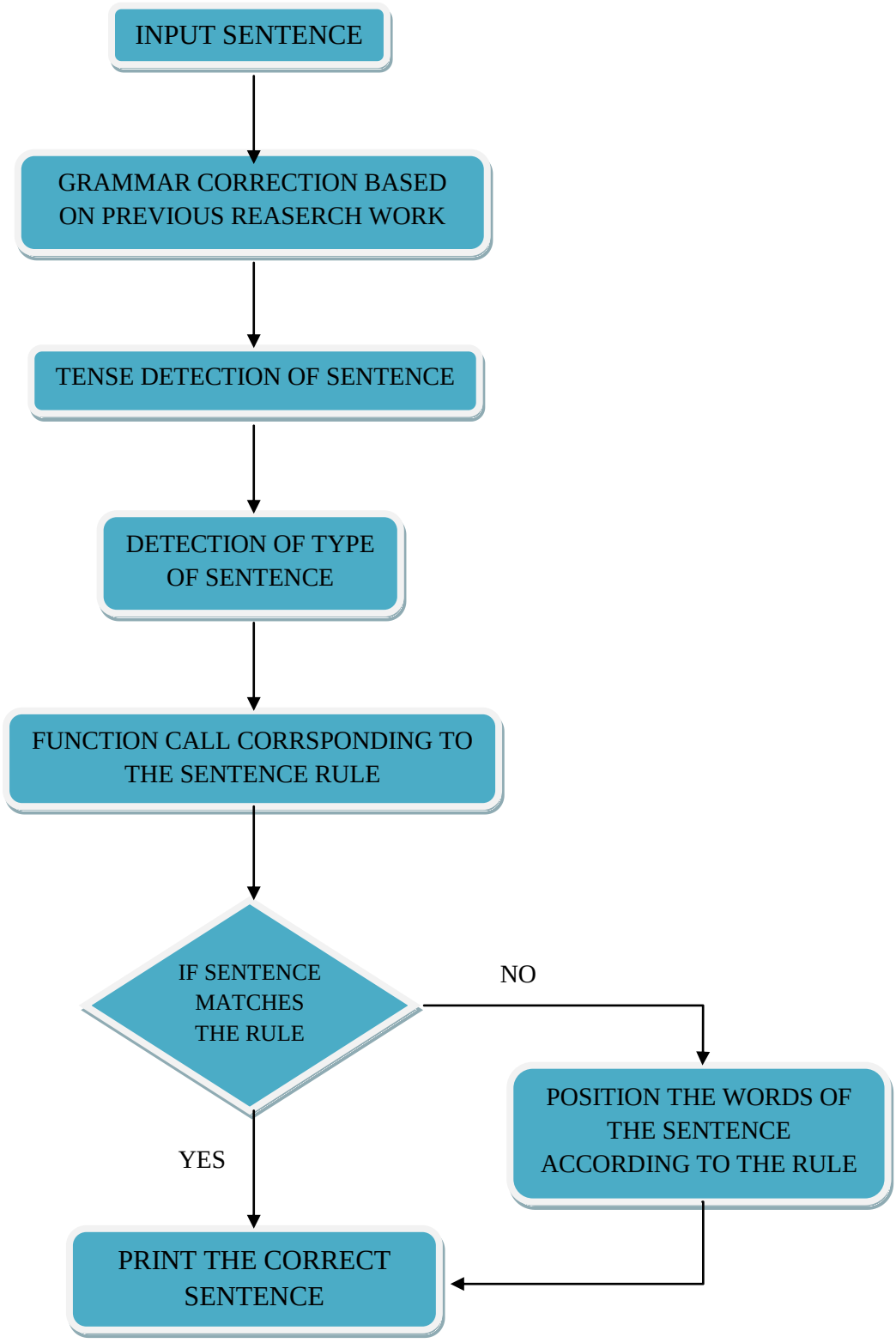

Fig. 1: System Flowchart

4.2 Algorithms Proposed

The algorithm 1 determines the tense of a sentence. For this we have used tense rules shown in figure 2.

	Present	Past	Future
Simple	Sub + 1st Verb + Object	Sub + 2nd Verb + Object	Sub +will/shall+ 1st Verb
Continuous	Sub + is/are/am + 1st Verb + ing + Object	Sub + was/were + 1st Verb + ing + Object	Subject + will/shall + be + 1st Verb + ing + Object
Perfect	Sub + has/have + 3rd Verb + Object	Sub + had + 3rd Verb + Object	Sub + will/shall + have + 3rd Verb + Object
Perfect Continuous	Sub + has/have + been + 1st Verb + ing + Object	Sub + had + been + 1st Verb + ing + Object	Sub + will + have + been + 1st Verb + ing + Object

Fig. 2: Tense Rules

To detect the tense of the sentence, we have computed a numeric value according to the verbs and helping verbs used in each tense. We have made an array coanating 13 rows, each row coantining a verb or a helping verb shown in the table 1.

Table 1: Helping verb array

Array Index	Verb/Helping Verb
0	Base form of verb
1	Second form of verb
2	Third form of verb
3	Verb+ es/s
4	Verb+ing
5	Will/shall
6	Has/have
7	Be
8	Was/were
9	Had
10	Have
11	Will
12	Been

We take an input sentence, and then we check for the verbs and helping verbs that are present in the sentence. We have a predefined array containing verbs and helping verbs. For the verb or helping verb present in the sentence, we put a value 1 on the index of that verb or helping verb in the predefined array. The example shown below exlapins this.

Example

1. Sadhana goes to college.
2. Sadhana had been going to college.

Table 2: Sentence array values

Array Index	Value
0	0
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0

Array Index	Value
0	0
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0

There is only one verb present in the first sentence, “goes”. It is of the form “Verb + s/es”. So value of array index is changed from 3 to 1 for this sentence.

We have three elements from the array in second sentence – had, been, and going. Therefore, the values at indices 9, 12, and 4 are set to 1.

The array of the two sentences will look as follows.

Table 3: Sentence array values filled

Array Index	Value
0	0
1	0
2	0
3	1
4	0
5	0
6	0
7	0
8	0
9	1
10	0
11	0
12	0

Array Index	Value
0	0
1	0
2	0
3	0
4	1
5	0
6	0
7	0
8	0
9	1
10	0
11	0
12	1

The array indices that having the value 1 are added, which gives a numeric value associated with both the sentences. The value corresponding to first sentence is 3 and for second sentence value is equals to the sum of 9, 12, and 4, i.e. 25.

Corresponding to each tense, we have generated a numerical value and stored it. Only a single value is associated with a tense. Table 3 represents the tense and the values associated to them.

Tense	Value
Simple Present	3
Present Continuous	4
Present Perfect	8
Present Perfect Continuous	22
Simple Past	9
Past Continuous	12
Past Perfect	11
Past Perfect Continuous	25
Simple Future	5
Future Continuous	16
Future Perfect	17
Future Perfect Continuous	37

Fig 3: Tense detection values

The values of the two sentences generated by algorithm are compared with the values of the figure 2. These values tell us the type of tense of the sentence. Value of first sentence is 3, and in the figure 2 value 3 corresponds to "Simple Present" tense, which implies first sentence is of simple present. Similarly, value 25 of sentence two matches the "Past Perfect Continuous" tense in figure 2, which means sentence two is of past perfect continuous tense. Algorithm 1 is the implementation of this procedure.

Algorithm 1: FIND-TENSE-OF-SENTENCE (sentence)

1 summ = 0, j=0

2 sent

3 create arrays verb_list[], tense_value[]

and tense_sum[]

4 **for** every word in sent

5 **if** sent[i] is a verb

```
6                got_verb[j] ← sent[i]
7                j = j + 1
8  for every word in verb_list
9        if verb_list[i] in got_verb
10               tense_value[i] ← 1
11 for every word in tense_value
12       if tense_value[i] == 1
13               sum ← sum + i
14 for i in tense_sum
15       if sum == tense_sum[i]
16                       print associated tense
```

Our work is divided into two parts – tense detection and sentence correction. We have designed two algorithms to handle these two parts separately. One is for determining the tense of the input sentence, and this is explained in algorithm ___. Second algorithm is for the correction of grammar and tense of the input sentence. Algorithm ___ describes this part.

Once we have determined the tense of the sentence, we pass the sentence to a function for correction. There are 12 different functions for this purpose, each function corresponding to a different tense, as each tense follows a different rule for correction. For example, if a sentence's tense is detected as "simple present", then it is sent to the function corresponding to simple present for its correction. The rules that are used to correct the sentence in the various functions are formed with the help of correct sentences belonging to the particular tense, so that we can correct as many sentences as we can. Consider the following examples:

Examples:

1) She was Laughing yesterday.

2) They were jumping on the hill.

3) A cat was walking on the roof.

The above mentioned examples are of past continuous tense, there can be so many examples. After detecting the type of tense, we need to identify the type of sentence to apply the correct rule for its correction.

With the help of these three example sentence, here we have explained the process of identification of sentence type. A sentence is basically a composition that contains various forms of words such as verb, noun, preposition, article, determiner etc. For determining the type of sentence, we have computed a score for each sentence. An array is used for this task.

Array Index	Word Form
0	Pronoun or Proper Noun
1	Base Verb
2	Noun
3	Preposition
4	Determiner
5	Article
6	2nd Verb
7	Auxiliary verb
8	Verb + ing

Fig. 4: Array generating score for sentence

For every input sentence, the word form of each word is checked. If the word form matches value 1 is added in the array. POS tagging is used to find the word form. For our three example sentences, figure 3 shows the corresponding arrays.

Table 4: Sentence score example

Array Index	Sentence 1	Sentence 2	Sentence 3
0	1	1	0
1	0	0	0
2	1	1	2
3	0	1	1
4	0	0	0
5	0	0	2
6	0	0	0
7	1	1	1
8	1	1	1
Score	17	20	32

Each tense has various types of sentences, and for each type of sentence a score is stored in an array. If the score of input sentence matches with the score of type of sentence stored in the array, then function corresponding to that sentence is called to correct the input sentence.

A function named **score_associated_function** in algorithm 2 is explained below.

Example:

The exam in June he passed.

Our system first determines the tense of the input sentence. The sentence in above example is detected of the tense simple past. Then system checks which sentence of simple past category matches with this sentence. Type of sentence is identified on the basis of a score. When we get the score computed by the algorithm, a function score_ssociated_function associated with it is called. The sentence in the example is incorrect according to the simple past tense rule. Correct rule is "*subject + 2ndverb + article + noun + preposition + noun*". The function places the words in their correct order on the basis of this rule. The output of which is "He passed the exam in June".

Algorithm 2: Correct Sentence

```
1  score ← 0
2 sent
3  create an array word_form and stored_score
4  for i in sent
5        if sent [i] == pronoun or proper noun
6              word_form[2] ++
7        if sent [i] == 2nd verb
8              word_form[1] ++
9        if sent [i] == noun
10             word_form[2] ++
11       if sent [i] == preposition
12             word_form[3] ++
13       if sent [i] == determiner
14             word_form[4] ++
15       if sent [i] == article
16             word_form[5] ++
17       if sent [i] == base verb
18             word_form[6] ++
 19      if sent[i] == auxiliary verb
20             word_form[7] ++
```

```
21      if sent[i] == verb + ing
22              word_form[8] ++
23  for i in word_form
24      score ← score + ( i * word_form[i] )
25  if score in stored_score[]
26      call corresponding score_associated_function
```

Grammar checking tools available in the market are less of value as they promise, and only very few tools have done an extensive study. We have made a tool which is combination of the previous research work and what we have proposed in this paper i.e word order correction. We have taken 9 grammar checking tools and compared their result with the result of our tool. In order to compare the tools we have taken 36 incorrect sentences and checked for errors on all the tools. Then we calculated the accuracy of all the tools and found that accuracy of our tool was the maximum.

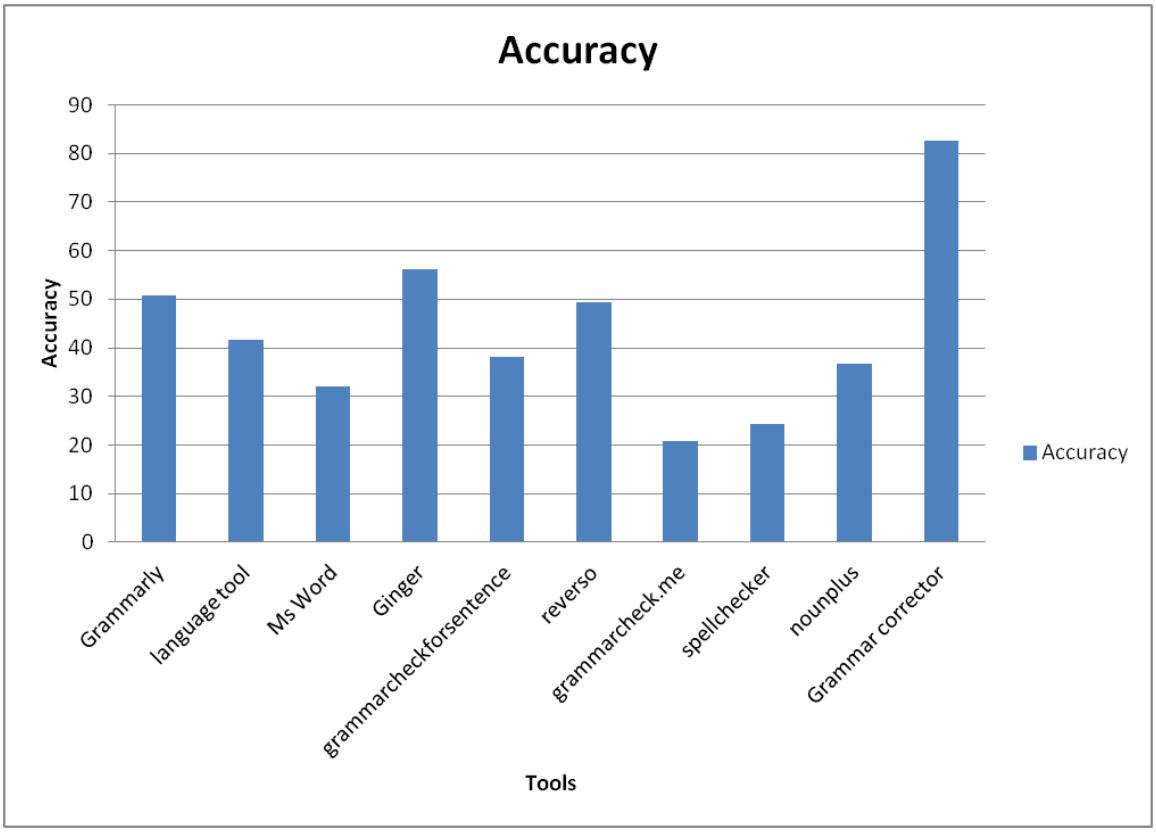

Fig. 4: Accuracy

Below we have explained the results of 36 sentences on top best tools according to their accuracy

1. In this 1st sentence there is error of word ordering, the incorrect sentence is ”At break-fast eats bread Ram” and the correct sentence is ”Ram eats bread at breakfast”, only ”Grammar corrector” was able to detect and correct this error.
2. The 2nd incorrect sentence is ”Krishna write a letter to Ram” and the correct sentenceis ”Krishna writes a letter to Ram”. All the four grammar checking tool successfully detected and corrected the error.
3. The 3rd incorrect sentence is ”Played football is by Ram” and the correct sentence is ”Football is played by Ram”, only ”Grammar corrector” was able to detect and correct the error.
4. The 4th incorrect sentence is ”I is studying English” and the correct sentence is ”I am studying English”. All the four grammar checking tools were able to detect and correct the error.
5. The 5th incorrect sentence is ”I am a book reading” and the correct sentence is ”I am reading a book”, only ”Ginger” and ”Grammar corrector” were able to detect and correct the error.
6. The 6th incorrect sentence is ”She are listening to the news” and the correct sentence is ”She is listening to the news”, all the four grammar checking tools were able to detect and correct the error.
7. The 7th incorrect sentence is ”The guests has arrived” and the correct sentence is ”The guests have arrived”, all the four grammar checking tools were able to detect and correct the error.
8. The 8th incorrect sentence is ”My all tasks I have finished” and the correct sentence is ”I have finished my all tasks”, only ”Grammar corrector” was able to detect and correct the error.
9. The 9th incorrect sentence is ”The work has not completed been by him” and the correct sentence is ”The work has not been completed by him”, ”Ginger” and ”Reverso” were able to detect that there is an error but they were not able to detect the correct error. ”Grammar Corrector” was able to detect and correct the error successfully.

10. The 10th incorrect sentence is ”He has been studying in these school” and the correct sentence is ”He has been studying in this school”, only ”Ginger” and ”Reverso” were able to detect and correct the error.
11. The 11th incorrect sentence is ”Living in New York he has been” and the correct sentence is ”He has been living in New York”, only ”Grammar corrector” was able to detect and correct the error.
12. The 12th incorrect sentence is ”Had he being using the car” an the correct sentence is ”He had been using his car”, ”Ginger”, ”Grammarly” and ”Reverso” are able to detect and correct the error. ”Grammar Corrector” neither detects nor corrects the error.
13. The 13th incorrect sentence is ”Amit played cricket on morning with Ram” and the correct sentence is ”Amit played cricket in morning with Ram”, none of the grammar checking tools were able to detect this error.
14. The 14th incorrect sentence is ”Cricket with Ram I played” and the correct sentence is ”I played cricket with Ram.”, only ”Grammar corrector” was able to detect an correct the error.
15. The 15th incorrect sentence is ”Played cricket I” and the correct sentence is ”I played cricket”, ”Grammar corrector” was able to detect and correct the error. ”Ginger” and ”Grammarly” detected some error but they were not able to detect the correct error.
16. The 16th incorrect sentence is ”He were laughing at me” and the correct sentence is ”He was laughing at me”. All the grammar checking tools were able to detect an correct the error.
17. The 17h incorrect sentence is ”At Ram laughing he was” and the correct sentence is ”He was laughing at Ram”, ”Grammar corrector” was able to detect and correct the error. ”Ginger” and ”Reverso” detected some error but they were not able to detect the correct error.
18. The 18 incorrect sentence is ”Was wearing she a pink dress” and the correct sentence is ”She was wearing a pink dress”, ”Grammar corrector” was able to detect and correct the error. ”Grammarly” detected some error but they were not able to detect the correct error.

19. The 19th incorrect sentence is ”He have lost my camera” and the correct sentence is ”He has lost my camera”, all the grammar checking tools were able to detect an correct the error.
20. The 20th incorrect sentence is ”In the exam appeared he had” and the correct sentence is ”He had appeared in the exam”, only ”Grammar corrector was able to detect and correct the error”.
21. The 21st incorrect sentence is ”The game not had been won by them” and the correct sentence is ”The game had not been won by them”. ”Ginger”, ”Reverso ” and ”Grammar Corrector” were able to detect and correct the error. ”Grammarly” detected some error but they were not able to detect the correct error.
22. The 22nd incorrect sentence is ”She have been waiting for you” and the correct sentence is ”She had been waiting for you” all the grammar checking tools were able to detect an correct the error.
23. The 23rd incorrect sentence is ”For you waiting she had been”, ”Grammar corrector was able to detect and correct the error” and ”Ginger” was able to detect the correct error but wasn’t able to correct the error.
24. The 24th incorrect sentence is ”He been has waiting for two days” and the correct sentence is ”He has been waiting for two days”, ”Grammarly”, ”Reverso” and ”Grammar corrector” were able to detect and correct the error.
25. The 25th incorrect sentence is ”They writing will been have essay” and the correct sentence is ”They will have been writing essay”, ”Grammar corrector” was able to detect and correct the error, ”Grammarly”, ”Ginger” and ”Reverso” were able to detect the error but were not able to detect the correct error.
26. The 26th incorrect sentence is ”Will loving her she have been” and the correct sentence is ”She have been loving her”, ”Grammar corrector” was able to detect and correct the error and ”Ginger” was able to detect the correct error but wasn’t able to correct the error and ”Reverso” detected some error but was not able to detect the correct error.
27. The 27th incorrect sentence is ”She will not have been live here” and the correct sentence is ”She will not have been living here”, ”Ginger” was able to detect and

correct the error and ”Grammarly” was able to detect the correct error but wasn’t able to correct the error.

28. The 28th incorrect sentence is ”He will have take the rest” and the correct sentence is ”He will have taken the rest”, ”Ginger” was able to detect and correct the error and ”Grammarly” detected some error but was not able to detect the correct error.
29. The 29th incorrect sentence is ”He have would finished the task” and the correct sentence is ”He would have finished the task”, ”Grammar corrector” was able to detect and correct the error. All other three detected some error but was not able to detect the correct error.
30. The 30th incorrect sentence is ”She will started have a new job” and the correct sentence is ”She will have started a new job”, ”Grammarly”, ”Ginger” and ”Reverso” were able to detect the correct error but wasn’t able to correct the error, ”Grammar corrector” detected some error but was not able to detect the correct error.
31. The 31st incorrect sentence is ”Coming here he will be” and the correct sentence is ”He will be coming here”, ”Grammar corrector” was able to detect and correct the error, ”Grammarly” was able to detect the correct error but wasn’t able to correct the error.
32. The 32nd incorrect sentence is ”He will not be coming their” and the correct sentence is ”He will not be coming here”, only ”Grammarly” was able to detect the correct error but wasn’t able to correct the error.
33. The 33rd incorrect sentence is ”He coming here will be” and the correct sentence is ”He will be comin here”, ”Grammar corrector” was able to detect and correct the error, ”Ginger” and ”Reverso” were able to detect the correct error but wasn’t able to correct the error.
34. The 34th incorrect sentence is ”She will got admission in a new school” and the correct sentence is ”She will get admission in the new school”, all the grammar checking tools were able to detect an correct the error.

35. The 35th incorrect sentence is ”Will get admission she in a new school” and the correct sentence is ”She will get admission in a new school”, only ”Grammar corrector” was able to detect and correct the error.
36. The 36th incorrect sentence is ”She will get admission on a new school” and the correct sentence is ”She will get admission in a new school”, ”Grammarly”, ”Ginger” and ”Reverso” were able to detect and correct the error.

Chapter 6 CONCLUSION AND FUTURE WORK

6.1 Conclusion

In this thesis, a word order corrector and a grammar checker have been developed. They have been tested on many types of sentences with errors. A set of rules has been developed which detects several types of errors. If the words of a sentence are not in a correct order, the sense of the sentence will be hard to understand, thus correcting the word order of a sentence is very important.

Our tool successfully corrected the sentences with the incorrect order of word sequence, which none of the other tools mentioned were able to do. Some types of grammatical errors were not solved by our tool that a few grammar checkers solved. However, the overall performance of our grammar checker is better as compared to other tools.

6.2 Future Work

We have tried to cover as many sentences as we could, however there are many types of sentences that are not covered. Hence we can extend this work by designing an algorithm which on encountering a new type of sentence that is not mentioned in our rules, will create a new rule for that sentence.

REFERENCES

[1] Tesfaye, Debela. “A Rule-Based Afan Oromo Grammar Checker”. Jimma Institute of Technology. Ethiopia: Vol. 2, No. 8, 2011

[2] Daniel Naber. “A Rule-Based Style And Grammar Checker”. Diplomarbeit. Technische Fakultät Bielefeld, 2003.

[3] Henrich, Verena; Reute, Timo. “LISGrammarChecker: Language Independent Statistical Grammar Checking”. Hochschule Darmstadt & Reykjavík University: 2009.

[4] “Support Materials and Exercises for GRAMMAR: PART I. Parts of Speech”. English Curriculum Content Expert, New Brunswick Community College: 1998.

[5] Singh, Mandeep; Singh, Gurpreet; Sharma, Shiv. “A Punjabi Grammar Checker”. Punjabi University. 2 nd international conference of computational linguistics: Demonstration paper. 2008. pp. 149 – 132

[6] Jahangir Md; Uzzaman, Naushad; Khan, Mumit. “N-Gram Based Statistical Grammar Checker For Bangla And English”. Center for Research On Bangla Language Processing. Bangladesh, 2006.

[7] Getachew , Mesfin. “Automatic Part of Speech Tagging for Amharic Language: An Experiment Using Stochastic Hidden Markov (HMM) Approach”. Master Thesis at School of Information Studies for Africa. Addis Ababa. Ethiopia, 2001.

[8] Gamback, Bjorn; Olsson, Fredrik; Alemu, Atelach, Lars Asker. “Methods for Amharic Part-of-Speech Tagging”. Proceedings of the EACL 2009 Workshop on Language Technologies for African Languages – AfLaT 2009, Athens. Greece, 31 March 2009. PP 104–111.

[9] S., Philip; M.W., David. “A Statistical Grammar Checker”. Department of Computer Science. Flinders University of South Australia.South Australia, 1996.

[10] Tsuruga, Ikki-machi, Aizu-Wakamatsu. “Dependency-Based Rules for Grammar Checking with LanguageTool”. Maxim Mozgovoy. University of Aizu. IEEE. Japan, 2011.

[11] Domeij, Rickard; Knutsson, Ola; Carlberger, Johan; Kann, Viggo. “Granska: An efficient hybrid system for Swedish grammar checking”. Proceedings of the 12th Nordic conference in computational linguistic, Nodalida- 99. 2000.

[12] Kinoshita, Jorge; Nascimento, Laиs do; Dantas ,Carlos Eduardo. ”CoGrOO: a BrazilianPortuguese Grammar Checker based on the CETENFOLHA Corpus”. Universidade da Sгo Paulo (USP), Escola Politйcnica. 2003.

[13] Karen Jensen, George E. Heidorn, Stpehen D. Richardson (Eds.): *Natural language processing: the PLNLP approach*, Kluwer Academic Publishers, 1993

[14] Fouvry, F. Constraint Relaxation with Weighted Feature Structures. In Proc. 8th International Workshop on Parsing Technologies. Nancy, France.2003

[15] Bolioli, A., Dini, L. & Malnati, G. JDII: Parsing Italian with a Robust Constraing Grammar. In Proc. COLING. Nantes, France. . 1992

[16] Lee, John Sie Yuen. *Automatic correction of grammatical errors in non-native English text*. Diss. Massachusetts Institute of Technology, 2009.

[17] Michaud, L., McCoy, K. F. & Pennington, C. A. An Intelligent Tutoring System for Deaf Learners of Written English. In Proc. 4th International ACM SIGCAPH Conference on Assistive Technologies (ASSETS). Arlington, VA. 2000

[18] Foster, J.. Parsing Ill-Formed Text using an Error Grammar. In Proc. Artificial Intelligence/Cognitive Science Conference (AICS) (pp. 55–60). Dublin, Ireland. 2003

[19] Bender, E. M., Flickinger, D., Oepen, S., Walsh, A. & Baldwin, T. rboretum: Using a Precision Grammar for Grammar Checking in CALL. In Proc. InSTIL/ICAL Symposium: NLP and Speech Technologies in Advance Language Learning Systems. Venice, Italy. 2004

[20] Copestake, A. Implementing Typed Feature Structure Grammars. Stanford, CA: CSLI Publications. 2002

[21] Izumi, E., Uchimoto, K., Saiga, T., Supnithi, T. & Isahara, H. Automatic Error Detection in the Japanese Learners' English Spoken Data. In Proc. ACL. Sapporo, Japan. 2003

LIST OF PUBLICATIONS

[1] K.K.Giri and R.K.Teckchandani "Speech to Text Conversion and Spell Correction using Bayesian Rule and Minimum Edit Distance", in Proceedings of *ISRD – International Conference of Current Research, 2017.***[Accepted]**