An Efficient Grammar Checking System by Using Shallow Parser

Thandar Htay, Myint Myint Khaing
Computer University, Pin Long, Myanmar
thandarhtay.tdh@gmail.com, myintkhaing06@gmail.com

Abstract

Grammar checking is one of the most widely used tools within natural language processing. Many word processing systems today include grammar checker which can be used to point out various grammatical problem in a text. This paper proposed a grammar checking system by using shallow parser. This proposed system consists of five main parts. Firstly, it can enter any sentence and then this sentence is split into words such as tokens in second step. Each token is identified into Part-of-Speech (POS) tag in third step. In step four, the system can analyze sentence by using shallow parser. Finally, it can display correct grammatical sentence by grammar checker. This system provides description about the grammar checking software developed for detecting the grammatical error on English texts and provides suggestions wherever appropriate to rectify those errors. A sentence does not have completed to be checked, instead the software can check the text while it is being typed and give immediate feedback.

Keywords: Natural Language Processing, Shallow Parsing, Grammar Checking

1. Introduction

Grammar checking is one of the most common applications in natural language processing and it is mostly used in word processors and compilers. Most of the word processing systems available in the market incorporate spelling, grammar and style-checking systems for other English and other foreign language, one such rule-based grammar checking system for English is discussed in [6]. Parser is one of the key components in Machine Translation (MT) system. To develop an efficient parser, almost complete sets of grammatical rules containing phrase structure rules as well as sentence structure rules have to establish.

Grammar checker needs a parser (or) syntax analyzer. Syntax analyzer determines whether a sting of tokens can be generated by a grammar. In this system, shallow parser method is used for syntax analysis. Shallow Parsing is the task of recovering only a limited amount of syntactic information from Natural Language (NL) sentences. Shallow parser has proved to be a useful technology for written and spoken language domain [1]. Shallow parser is a division of words that together consist of a grammatical unit.

The rest of the paper is organized as follows: related work is proposed in section2. Section 3 represents grammar analysis and phrase chunking with shallow parser. Section 4 describes design and implementation of the system works with shallow parser. Section 5 represents performance evaluation of this system. Finally, conclusion gives in section 6.

2. Related Work

Granska Swedish Grammar Checking is proposed using rule-based methods and error-correction methods. It implemented a hybrid system that combines probabilistic and rule-based methods to achieve high efficiency and robustness in Swedish grammar [3]. Tin Muyar Win, Zin Mar Than described rule-based chunk level grammar checking with parallel approach and it involves tokenization, rules-based tagging, and phrase chunking with chunk level grammar checking and clause segmentation [8]. Daniel Naber described about a rule-based style and grammar checker that it takes a text and returns a list of possible errors. To detect errors, each word of the text is assigned its part-of-speech tag and sentence is split into chunk, such as noun phrase, verb phrase etc. Then the text is matched against all the checker's pre-defined error rules. If the rule matches, the text is supposed to contain an error at the position of the match. The rules describe errors as patterns of words, part-of-speech tags and chunks [2].

3. Grammar Analysis

Grammar analysis is one of the natural language processing using a special NLP program that converts poor writing to acceptable writing. Special grammar analyzer programs can be used to process business communications and improve them.

Grammar is traditionally subdivided into inter-related studies: Morphology and Syntax.

Morphology is the study of how words are formed out of smaller units called morphemes. Syntax is the broader study of how words are strung together to form phrases, clauses, and sentences [4]. Syntax is concerned with how words are strung together to expressions from larger units of. In this paper, the grammar analyzer breaks each sentence down into parts of speech and shows the relationships among words using rules-based lexicon, part-of-speech tagging process, tokenizing and syntax analysis.

3.1. Tokenization

Tokenization is the process of breaking a sequence into words, punctuations and other symbols. These words and expression sequences are called tokens, and the tools performing such tokenization are tokenizer. The following example illustrates the basic function of a tokenizer.

Sample input sentence: This is a test.

It gets each token after tokenizing input sentence as shown in Figure 1.

This	is	a	test	.

Figure 1. Each token for sample sentence

Tokenizer breaks a given text into small pieces by delimiting at both white spaces and punctuations. It is possible to reassemble these tokens to larger lexical items.

3.2. Part-of-Speech Tagging

Words are divided into different kind or classes, called Parts of Speech (POS). POS tagging identifies the words in a given sentence corresponding to their parts of speech [2]. These tags depend on definition and context. POS tagger has two type of tagger; simple tagger and rule-based tagger. Simple tagger accepts the equal number of words tokens. Find their words in the lexicon. Then assign all possible tags for each word. Rule- based tagger can give multiple possible tags. But some are more likely than other.

There are eight parts of speech. They are noun, verb, pronoun, adjective, adverb, preposition, conjunction and interjection.

Grammar units		**POS tagging**
1.	Noun	<NN>
2.	Verb	<VB>
3.	Pronoun	<Pro N>
4.	Adjective	<JJ>, <ADJ>
5.	Adverb	<ADV>
6.	Preposition	<PRP>
7.	Conjunction	<COJ>
8.	Interjection	<IJ>

In English, many words can be used both as noun and as verbs. Although there are eight parts of speech in English grammar, seven are used for pattern rules in this system. They are noun, verb, pronoun, adjective, adverb, preposition and conjunction. POS tagging is thus a typical disambiguation problem: all possible tags of a word are known and the appropriate one for a given context needs to be chosen in figure 2.

He	Saw	the	Big	dog
Pro N	VBD	DT	JJ	NN

Figure 2. Tagging tokenized words to POS

3.3. Syntax Analysis

Syntax analyzer captures structure relationships between words and phrases. Grammar rules are used to describe the correct syntax of a language. Syntax analyzers assign a syntactic structure to a phrase on the basis of grammar. A syntax analyzer is also called a parser. There are many methods of parser in this syntax analyzing step as shallow parser, full parser, and chart parser, etc.

Shallow parser attempts to provide some machine understanding of the structure of a sentence. Shallow parsing involves identification of constituents of a sentence. A chunker is a division of the text's sentence into series of words that together consist of a grammatical unit [9]. Shallow parsing is also called partial parsing and then most often to the task of chunking. These methods can do firstly segmentation and labeling sequences of characters and then chunking.

3.3.1. Phrase chunking with shallow parser. Phrase chunking assigns a tag to word sequence. Typical chunks are noun phrase (NP) and verb phrase (VP).

3.3.1.1 Noun Phrase. In grammar, a noun phrase (NP) is a phrase whose head is a noun or pronoun, optionally accompanied by a set of modifiers. Noun phrase typically consist of determiner, adjectives and noun or pronoun as shown in Figure 3.

3.3.1.2 Verb phrase. A verb phrase (VP) is a syntactic structure composed of the predicative elements of a sentence and functions in providing information about the subject of the sentence. The verb phrase is a phrase headed by a verb [7]. A verb may be constructed from a single verb. The verb phrase will consist of various combinations of the main verb and auxiliary verbs, complements and adjuncts.

3.3.2. Shallow Parsing Algorithm. Invariantly of the software of algorithm chosen, a chunker ultimately has rules that decide where do different phrase chunk begin and end depending on the text and POS tag. POS tagging techniques can usually be adapted for chunking. A common method is to consider the beginning and continuation of chunk as special types of tags. To chunk a sentence into NP, VP and PP chunks that might have the following tags:

B-X
B word begins a chunk of type X (NP, VP, PP, and so forth).
I-X
I word belongs to a chunk of type X but does not begin it.
O
O word does not belong to any chunk.

An O tag would not be used where full chunking is needed, because every word would belong to some chunk [5].

3.4. Generating Grammatical Sentence

Generating grammatical sentence is displayed the correct sentence for output. Then error message is given by the grammar checker when there is no in fact no error in the text. Rule-based checking is a set of rules which is matched against a text which has at least been POS tagged. A set of rules describes how sentence are built. Grammar checker uses context of a sentence. It works on complete sentences but not on single word. Grammar checking rules that will identify the correct sentence by matching rules in the rules directory which depend on POS tags. The pattern is a sequence of words, POS tags or chunk. If this sequence pattern is found in pattern matching, it can be declared as a correct sentence. This system used thousands of rules. The sample grammar checking rules are as follows:

NN V JJ PRP NN
NN V JJ PRP POS NN
NN V JJ PRP POS NN PRP NN
NN V PRP DT NN
NN V POS JJ NN
POS NN V ADV PRP NN
POS NN V JJ PRP POS
POS NN V NN NNS
POS NN V Pro N NN
Pro N V PRP ADV
Pro N V PRP DT NN
Pro N V PRP POS NN
Pro N V ProN ADV ADV DT NN
Pro N V ProN DT NN PRP NN
Pro N V POS NN PRP NN
Pro N V JJ DT NN PRP Pro N
DT NN V ADV JJ
DT NN V JJ NN
DT NN V JJ PRP NN
ADV V DT NN
ADV V DT NN PRP DT NN
ADV JJ PRP DT NN V ADV, etc.

4. Proposed System Architecture

The system works by splitting the sentence into words that can be checking by using simple sentence English grammar rules as shown in Figure 3.

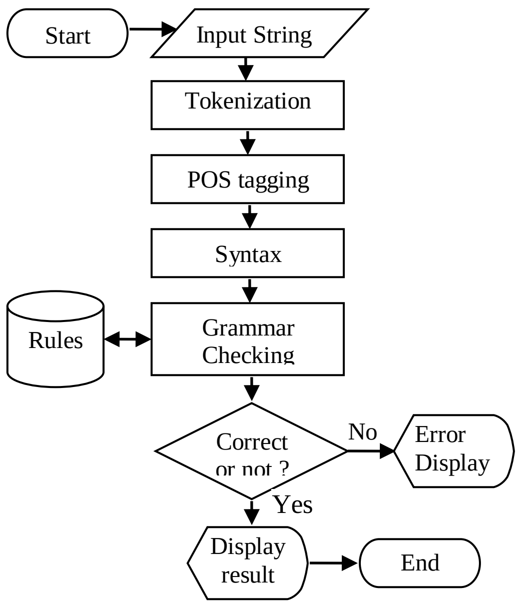

Figure 3. Proposed System Architecture

In this proposed system, first step is entering a grammatical sentence or ungrammatical sentence into the system. And then the system tokenizes input sentence which is spilt into "token" as words and punctuations: e.g., white space, punctuation, full stop, etc. And then the tokenizing words corresponding to their parts of speech. This step is syntax analyzing by using shallow parser. It captures structure relationships between words and phrases. The tokenized sentence is divided into phrase chunks; noun phrase as show in Figure 4 and verb phrase, etc.

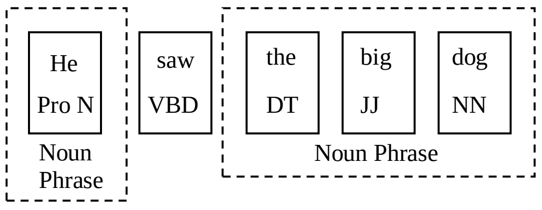

Figure 4. Noun phrase chunking with shallow Parser

Noun Phrase Chunking
NP <DT><JJ><NN>
NP <DT><JJ><NN><NNS>
NP <DT><JJ><NN><NNS><PRP><POS>
NP <Pro N><JJ><NN><NNS>
NP <Pro N><JJ><NN>
NP <PRP><JJ><NNP><NN><NNS>

NP	<PRP><POS><NN>
NP	<POS><JJ><PRP><DT><NN>
NP	<JJ><PRP><NN>
NP	<JJ><DT><NN><PRP><Pro N>, etc.

Verb Phrase chunking

VP	<V>
VP	<V><NP>
VP	<V><PP>
VP	<V><PRP><NP>, etc.

Finally, grammar checker is designed with error detection rules to detect potential errors in the sentence and provide correction to resolve those errors. It works on complete sentences but not on single word. Checking rules identify the correct sentence by matching rules in the rules dictionary which depend on POS tags. The most important part of a rule is pattern. If this pattern is found in pattern matching, it can be displayed as a correct sentence. For example,

Input sample sentence: He saw big a dog.

Tokenization:

He	saw	big	a	dog	.

POS Tagging: <Pro N>He<VBD>saw<JJ>big <DT>a<NN>dog

Phrase chunking: (B-NP) He (I-VP) saw (B-NP) big (I-NP) a (I-NP)dog(O).

Grammar generation: He saw a big dog.

5. Performance Evaluation

In measuring precision and recall, precision is the ratio of number of patterns which are correctly patterns to the number of proposed patterns. And recall is the ratio of patterns which are correctly tagged to the number of total patterns. The result of grammatical patterns on testing rules as shown in Table 1.

$$\text{Precision} = \frac{\text{Number of correct patterns}}{\text{Number of proposed patterns}} \text{X } 100\ \%$$

$$\text{Recall} = \frac{\text{Number of correct patterns}}{\text{Number of total patterns}} \text{X } 100\ \%$$

Table 1. Result of grammatical patterns on testing rules

Proposed patterns	3569
Correct patterns	3489
Total patterns	3650
Precision	97.75
Recall	95.58

5.1. Performance Speed of the system

A performance of grammar checking system by using shallow parser is written in java by using hand written grammar rules and classification rules. The performance speed of this system is measured by execution time in terms of word and second. Execution time is measured on a system of Pentium IV, processor speed 3.20 GHZ and Random Access Memory 256 MB. Execution time is measured as the time taken from getting input sentence to generating analyzed sentence. That means it is time spent for the entire grammar checking system execution. Each word count level is executed for twenty times and calculates for that word count level. Word count and corresponding average time for simple sentence structure is shown in Table 2 and Figure 5.

Table 2. Execution time and corresponding word count for simple sentence

Words	Second
25	3.3451
50	4.3546
75	4.6452
100	6.775
125	7.6972
150	9.0641
175	9.5269
200	11.0609

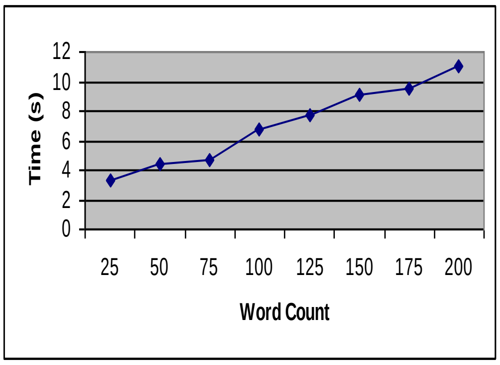

Figure 5. Execution time and corresponding word count for simple sentence

6. Conclusion

This system can perform tokenizing, POS tagging, phrase chunking with shallow parsing approach. It can check the whole sentence whether they are grammatically correct or not. This system can be built incrementally, starting with just one rule and then extending it rule by rules. As the operations are performed, then the time taken by a computation can be significantly reduced. This system can check the simple sentences according to the pattern rules. It can check the sentence spelling error and tenses of sentence. But the system can't be able to rewrite error words. The result of the system can be used not only in grammar analyzer weather it is rule-based or shallow parsing but also in machine translation system, summarization and information extraction system

References

[1] B. Meryesi, "Shallow parsing with POS taggers and Linguistic Knowledge", Journal of Machine Learning Research, 2002.

[2] D. Naber, "A Rule-Based Style and Grammar Checker", 28 August 2003.

[3] J.B. Viggo, "Granska Grammar Checker for Swedish Second Language Learners" [online document].

[4] J.Galasso, "Analyzing English Grammar: An Introduction to Feature Theory", 2002.

[5] L.A. Ramshaw and M.P. Marcus,"Text chunking using transformation-based learning", In Workshop on Very Large Corpora, 1995.

[6] M.S. Gill, G.S. Lehal, "Punjabi Grammar Checker", 2008.

[7] P.C. WREN and H. MRTIN, "High School Grammar and Composition", S.Chand & Company Ltd, Hundred and Seventeenth Edition, 1986.

[8] T.M. Win, Z.M. Than, "Rule-based Chunk Level Grammar Checking", 2008.

[9] S. Abney, "Parsing by Chunk", Kluwer Academic Publisher, Dordrecht, 1991.