Chunk Tagged Corpus Creation for Myanmar Language

Phyu Hninn Myint, Tin Myat Htwe and Ni Lar Thein
University of Computer Studies, Yangon, Myanmar.
phyuhm@gmail.com

Abstract

In the applications of Natural language processing (NLP), sentence analysis is one of the important phases for machine translation systems. Currently, no mature deep analysis that has been worked done is available for Myanmar language. To perform shallow parsing on sentences, the chunk identification is a fundamental task. The POS tagged corpus creation has been proposed in [8] and in this paper, we have proposed a methodology for building chunk tagged corpus for Myanmar Language. We use the POS tagged corpus that is proposed in [8] and identify chunks in Myanmar POS tagged texts. Our approach uses rule-based on how to identify all chunks in a Myanmar sentence. As a preprocessing step, normalization of POS tags is needed to perform in order to produce finer tags. Hence, normalization rules are also developed. After normalization, chunk rules are applied to tag chunk for these finer tags. Our chunk tagged corpus is very useful in Myanmar to English machine translation system.

1. Introduction

In general, text chunking consists of identifying non-recursive phrase structures from a sequence of tokens and classifying them into some syntactic categories like base noun phrases and base verb phrases. As an intermediate step towards full parsing, text chunking has been attracting more and more attention in the NLP community. It is recognized as an important subtask of many large NLP applications such as machine translation, text mining and question answering. Chunking is the process of annotating tagged tokens with structures in a non-hierarchical and non-recursive way. Text chunking is a useful preprocessing step for parsing. It consists of dividing a text into phrases in such a way that syntactically related words become member of the same phrase. These phrases are non-overlapping which means that one word can only be a member of one chunk. In CoNLL-2000 chunking task, chunking was defined as "the task of dividing a text into phrases in such a way that syntactically related words become members of the same phrase" [9].

There are multiple approaches to solving NLP problems: Rule Based, Statistics Based, Hybrid systems, etc. Rule Based (Symbolic) system is developed like traditional expert systems by using hand coded rules. Advantages of rule based system are: it is fast to develop and it doesn't require large datasets. Disadvantage of that system is: it is costly to maintain. Statistics Based (Empirical) system annotate data based on standard tagsets, and then machine learn a model. Advantages are: it is a current trend and robust, and it performs better. Disadvantages are: it is extensive upfront cost, it requires lots of data and improvement may not correct obvious errors. Hybrid systems often blend rule-based pre- and post-processing with machine learning core. Human intuition plays a large role in both, either in coding the rules directly or in deciding what features to use. It can be driven by error analysis.

Since the early 90's, several techniques for carrying out shallow parsing have been developed. These techniques can also be classified into two main groups: based on hand-code linguistic rules and based on learning algorithms. These approaches have a common characteristic: they take the sequence of lexical tags proposed by a POS tagger as input, for both the learning and the chunking processes [1].

However, when developing a chunker for a new language, statistical and machine learning methods require an already preprocessed corpus which is not always available. In our case, such a corpus for Myanmar language does not exist at all so we had to opt for the rule-based approach that would, in the prospect, end up with a data-set usable for future testing of similar systems for Myanmar language. In this paper, we describe the attempt to build the rule-based chunker in order to develop the chunk tagged corpus for Myanmar language and evaluate its performance. The typical chunk consists of a single content word surrounded by a constellation of function words, matching a fixed template. A simple context-free grammar is quite adequate to describe the structure of chunks. Therefore, our rules are described in the context-free-grammar structure.

The rest of the paper is organized as follows: Section II describes the related work. Section III gives a brief description of our chunking methodology. Section IV describes our customized chunk tagset. Section V presents normalization rules. Section VI discusses about chunking rules. Finally, the experimental results and some conclusions on this work are given in section VII and section VIII respectively.

2. Related Work

Chunking is today considered to be the preprocessing stage that may facilitate the full parsing of sentences of a certain language. This task has already been proven using different methods: rule-based, memory-based, statistical and combined systems. However, when developing a chunker for a new language, statistical and machine learning methods require an already preprocessed corpus which is not always available.

In the early nineties, Steven P. Abney [1] proposed to approach parsing by starting with finding related chunks of words. Author said that a typical natural language parser processes text in two stages. A tokenizer/morphological analyzer converts a stream of characters into a stream of words, and the parser proper converts a stream of words into a parsed sentence, or a stream of parsed sentences. In a chunking parser, the syntactic analyzer is decomposed into two separate stages, which he called the chunker and the attacher. The chunker converts a stream of words into a stream of chunks, and the attacher converts the stream of chunks into a stream of sentences. Then, this author described that the chunker is a non-deterministic version of an LR parser. An LR parser is a deterministic bottom-up parser. It is possible to automatically generate an LR parser for any of a large class of context-free grammars. The parser shifts words from the input string onto the stack until it recognizes a sequence of words matching the right-hand side of a rule from the grammar. At that point, it reduces the sequence to a single node, whose category is given in the left-hand side of the rule.

Steven P. Abney [2] proposed that corpus-oriented computational linguistics is technique for bootstrapping broad-coverage parsers from text corpora. The work is a step along the way toward a bootstrapping scheme that involves inducing a tagger from word distributions, a low-level “chunk” parser from a tagged corpus, and lexical dependencies from a chunked corpus. Author presented a technique, finite-state cascades, for producing fast, robust parsers for unrestricted text. The technique has been applied to English and German, and is being used in a project for inducing sub-categorization frames and selected restrictions in these languages. The parser consists of a pipeline of finite-state recognizers. Key concepts are easy-first parsing, islands of certainty, and containment of ambiguity. Finite-state cascades can be extended to include feature assignment and output of “linguistic” structure at little cost in efficiency.

K. Vuckovic, M. Tadic and Z. Dovedan [10] presented the first attempt to develop a chunker for Croatian. They opted for rule-based approach since there are no corpora annotated for chunks in Croatian and no machine-learning or statistical-based methods could be applied. Obtained results showed correctly assigned chunk boundaries and types leading them to the conclusion that rule-based paradigm would be a reasonable choice for chunking of larger Croatian corpora in the future.

I. Boehm [3] compared two chunking approaches, namely an approach based on regular expression rules developed by a human and a machine based chunking approach based on a N-gram statistical tagger. Experimental results showed that the performance of the machine based chunker is very similar to the results obtained by the regular expression chunker.

3. Methodology

There are broadly two approaches for the development of chunkers - the linguistic approach which depends upon hand-crafted grammars, and the machine learning approach where chunkers are learned automatically from a labeled training corpus. Since there is no available chunk tagged corpus for Myanmar language, we use rule-based chunker to build chunk tagged corpus by applying available POS tagged corpus. In order to be able to parse a sentence, a defined set of rules, called a "grammar" is needed. For simple texts, a simple grammar often suffices, but as the complexity of texts increases, the size of the grammar increases too.

3.1. Context-Free Grammar

Groups of words may behave as a single unit or phrase, called a constituent. The most common way of modeling constituency is Context-Free Grammar (CFG). A CFG defines a formal language: the set of all sentences that can be derived by the grammar. Context free grammars are powerful enough to describe the syntax of most programming languages; in fact, the syntax of most programming languages is specified using context-free grammars.

A Context-Free Grammar is a 4-tuple (N,T,P,S) where N is a finite set of non-terminal symbols, T is a finite set of terminal symbols, disjoint from N, S is a special designated symbol from N called the Start Symbol, and P is a finite set of Production Rules (or Productions), of the form A -> α where A is any non-terminal symbol and α is any sequence of terminal and non-terminal symbols. These grammars are called 'context free' because all rules contain only one symbol on the left hand side and wherever we see that symbol while doing a derivation, we are free to replace it with the stuff on the right hand side.

A CFG is described briefly as follows:

G = <T, N, S, R>

- T is set of terminals
- N is set of non-terminals
- S is start symbol (one of the nonterminals)
- R is rules/productions of the form A -> α - where A is a nonterminal and α is a sequence of terminals and nonterminals (may be empty).
- A grammar G generates a language L.

In our approach, using CFG, in normalization step, terminals set includes basic tags from lexicon and non-terminals are finer tags. In chunking step, finer tags and basic tags become the members of terminal set and chunk tags are included in the non-terminals set.

3.2. Proposed Approach

The input to the parsing system is one sentence, either plain or POS tagged. Output is an ordered set of parses. The aim is to produce all possible parses in ranked order hoping to get the best parse to the top. In parsing, a sentence is a sequence of chunks. Chunks are sequences of words.

This paper presents about a system which develop a chunk tagged corpus from POS tagged corpus. We use the POS tagged corpus proposed by the system described in [8]. The input of the system is Myanmar sentences which are tagged with basic POS tags from this corpus. These tags are driven from Myanmar dictionary so that it can be called stem word tagging. These tags are needed to normalize using lexical rules. Output of this step is Myanmar sentences which are tagged with finer POS tags. Then, chunk rules are applied to chunk these sentences and to create chunk tagged corpus. The main steps of the system are as follows:

(A) Normalizing and forming finer tag
(B) Chunking and creating chunk tag

Figure 1 demonstrates the scheme of the system.

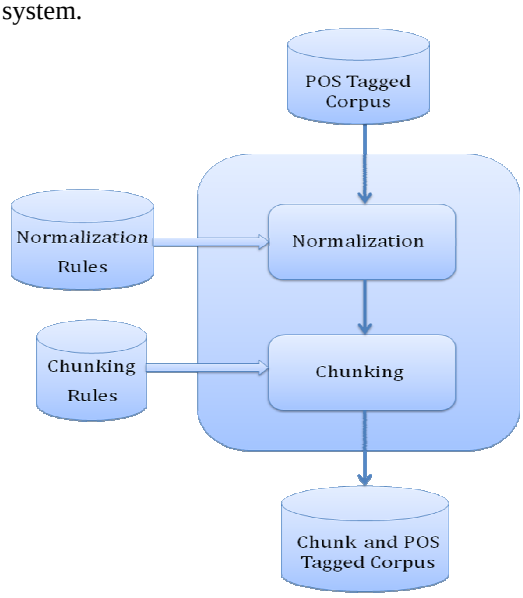

Figure 1. Chunk tagged corpus creation scheme

A. Normalizing and forming finer tag

We have done some preliminary experiments including corpus analysis to understand the nature of Myanmar language. Normalization step is needed to form more meaningful words and annotate with more appropriate finer POS tags and categories. In our language, Myanmar, there are many "Particles" in the text. These can be appeared in binding with Noun, Verb, Adjective and Adverb in the text. Moreover, these can convert the type of POS tag, that is, Noun attached with some particles can become Verb or Adjective. Also, Verb or Adjective with some particles can create new POS tag, which is Adjective with superlative or comparative degree. There are the same pattern and particle to transform from one POS tag to another. Therefore, some lexical rules have to be developed to deduce more finer and standard POS tag. The normalization rules are explained with the description of CFG and the detail examples in Section V.

B. Chunking and creating chunk tag

For chunking system, chunk rules have to be defined to assemble the POS tagged words. Finer POS tags have to be used to identify chunk. In addition, some particles are needed to combine with other POS tags in order to create one chunk. Therefore, chunking rules have to be developed and they are revealed with the description of CFG and how to chunk the sentences is described with detail examples in Section VI.

4. Customized Chunk Tagset

A chunk consists of a head word and its modifiers. The customized chunk tagset uses only 7 chunk tags. The customized basic POS tagset used in the input corpus includes only 14 POS tags. To obtain more accurate lexical information together with POS tag, category of a word has to be added by Myanmar grammar. This category can be applied in further NLP applications. The category for a word can be formed from the features of that word. For example, Noun has 16 categories such as Objects, Person, Animals, Food, Location, etc. For instance, "မိန်းကလေး" (girl) word must be tagged with NN.Person (Person category of Noun tag). Also, "သို့" (to) has to be tagged with PPM.Direction (Direction type of Postpositional Marker).

Moreover, finer POS tagset, including only 6 tags, is proposed to normalize the basic tags. Customized POS tagsets and chunk tagset are described in Appendix section at the end of the paper.

5. Normalization Rules

This paper creates lexical rules for finer POS tagging and using these rules, finer and standard POS tags can be produced. These finer tags are able to be applied in the later steps of NLP applications. It is possible that word with finer tag can be directly translated to other language. We have to analyze "Particles" which are functional words to develop most of the lexical rules.

In Myanmar language, there are many

particles which can be called affixes of the word and can cause the changes of sense or type of that word. The prefixes are "မ-"(ma-), "အ-"(a-) and "တ-"(ta-). The prefix "မ-" (ma-) is an immediate constituent of the verb, which is the head of the word construction as in: ma-swa: မ-သွား: 'not go'; ma-kaung: မ-ကောင်း : 'not good'. It changes the positive sense to negative sense of the word. The scope of verbal negation extends to the whole compound of a compound verb, as in ma-tang pra: မ-တင်ပြ : 'not submit'; ma-saung-ywat : မ-ဆောင်ရွက် : 'not carry out'. Another pattern of negation is possible with verb compounds or verb phrases by individualized negation of each portion of the compound, as in: ma-ip ma-ne : မ-အိပ် မ-နေ : 'not sleep at all'; ma-tang ma-kya: မ-တင် မ-ကျ : 'noncommittal'.

The prefix "အ-" (a-) is a type converter which is the head word of the verb or adjective as in: a-lote: အ-လုပ် : 'work or job'; a-hla : အ-လှ : 'beauty'. The prefix "တ-" (ta-) can also be seen as a type converter, as in ta-lwal ta-chaw: တ-လွဲ တ-ချော် : 'wrongly'.

The postfixes are "-မှု" (-mhu), "-ခြင်း" (-ching), "-ချက်" (-chat), "-ရေး" (-yay), "-နည်း" (-nee), "-စွာ" (-swar), "-သော" (-thaw), "သည့်" (-thi), "-မည့် " (-myi), etc. The postfixes "-မှု" (-mhu), "-ခြင်း" (-ching), "-ချက်" (-chat), "-ရေး" (-yay), "-နည်း" (-nee) change the type of the previous POS tag from verb or adjective or adverb to noun. The words ended with these postfixes are in the noun form. Also, the postfixes "-သော" (-thaw), "သည့်" (-thi), "-မည့် " (-myi) convert to the adjective form from adjective or adverb or verb. The postfixes "-စွာ" (-swar) alters the type of adjective or verb or adverb to form adverb. In noun form, the postfixes "-များ" (-myar), "-တို့" (-doh) change the singular noun to plural noun.

Moreover, in adjective, if JJ tag is lied between two affixes "အ" (-a) and "ဆုံး" (-sone), this tag JJ become to JJS (superlative degree), i.e., " အ JJ ဆုံး " is equal to "JJS".

Some of the normalization rules are described using CFG as follows ::

JJ -> JJ (သော | သည့် | မည့်)

JJS -> A JJ ဆုံး

JJC -> (ပို၍ | သာ၍ | ပို) JJ

RB -> RB စွာ

NNR -> NN (များ | တို့)

NNR -> PRN (များ | တို့)

NN -> A VB

NN -> VB (မှု | ခြင်း | ချက် | ရေး | နည်း)

The sample input text from the POS tagged corpus and output of the normalization step are as follows:

Example 1::

Before Normalization,

- " ကျန်းမာ/**VB.Common** # ခြင်း/**PART.Common** # သည်/PPM.Subject # လာဘ်/NN.Common # တစ်/PART.Number # ပါး/PART.Type # ဖြစ်/VB.Common # သည် /SF "

After Normalization,

- " ကျန်းမာခြင်း/**NN** # သည်/PPM.Subject # လာဘ် /NN.Common # တစ် /PART.Number # ပါး /PART.Type # ဖြစ်/VB.Common # သည် /SF "

Example 2::

Before Normalization,

- " သူ/**PRN.Person** # တို့/**PART.Number** # သည်/PPM.Subject # A တန်း/NN.Common # ထဲတွင်/PPM.Extract # A /**PART.Common** #တော်/**JJ.Demonstrative** # ဆုံး/**PART.Common** # ကျောင်းသား/**NN.Person** # များ/**PART.Number** # ဖြစ်/VB.Common # ကြ/PART.Support #သည်/SF "

After Normalization,

- "သူတို့/**NNR** # သည်/PPM.Subject # အတန်း/NN.Common # ထဲတွင်/PPM.Extract # အတော်ဆုံး/**JJS** # ကျောင်းသားများ/**NNR** # ဖြစ်/VB.Common # ကြ/PART.Support #သည်/SF "

6. Chunking Rules

For chunking steps, chunk rules have to define to build chunk for POS tagged words. There are many particles which are attached with finer POS tags. Most of them are postfixes of verb. They include -ခဲ့ (-khae`), -ကြ (-kya), -လေ (-lay), -နေ (-nay), -ထား (-htar), လိုက် (-lite), - ချင် (-chin), etc. We have to combine adjective and noun tags to form noun chunk and also adverb and verb tags to verb chunk.

Some of the chunking rules are described using CFG as follows ::

NC -> NN PART.Number

NC -> NN PART.Common PART.Type

NC -> PRN PART.Possessive NN

NC -> NN PART.Possessive NN

NC -> PRN PART.Number PART.Possessive NN

NC -> (JJ| JJS| JJR) NN

NC -> (JJ| JJS| JJR) NNR

NC -> PRN NN

NC -> PRN NNR

NC -> NN JJ

NC -> NNR JJ

NC -> NN NN* | NNR NNR* | NNP NNP*

VC -> VB PART.Support*

VC -> VB VB*

VC -> RB VB

RC -> RB | RBS | RBR

RC -> RB RB*

JC -> JJ | JJS | JJR

JC -> JJ JJ*

PPC -> PPM

SFC -> SF

After normalization, the finer POS tags are chunked at the chunking step as follows:

Example 1::

Before Chunking,

- " ၄င်္ကျား/NNR # သည်/PPM.Subject # ကောင်းကင်/NN.Natural # တွင်/PPM.Place # အုပ်စုဖွဲ့/VB.Common # ၍/CC.Sent # ပျံသန်း/VB.Common # နေ/Part.Support # ကြ/Part.Support # သည်/SF "

After Chunking,

- " **NC** [၄င်္ကျား/NNR] # **PPC** [သည်/PPM.Subject] # **NC** [ကောင်းကင်/NN.Natural] # **PPC** [တွင်/PPM.Place] # **VC** [အုပ်စုဖွဲ့/VB.Common] # **COC** [၍/CC.sent] # **VC** [ပျံသန်း/VB.Common, နေ/Part.Support, ကြ/Part.Support] # **SFC** [သည်/SF] "

Example 2::

Before Chunking,

- " သူတို့/NNR # သည်/PPM.Subject # အတန်း/NN.Common # ထဲတွင်/PPM.Extract # အတော်ဆုံး/JJS # ကျောင်းသားများ/NNR # ဖြစ်/VB.Common #ကြ/PART.Support #သည်/SF "

After Chunking,

- " **NC** [သူတို့/NNR] # **PPC** [သည်/PPM.Subject] # **NC** [အတန်း/NN.Common] # **PPC** [ထဲတွင်/PPM.Extract] # **NC** [အတော်ဆုံး/JJS # ကျောင်းသားများ/NNR] # **VC** [ဖြစ်/VB.Common # ကြ/PART.Support] # **SFC** [သည်/SF] "

7. Experimental Results

In order to measure the performance of the system, we have tested many experiments using our approach on different types of sentences till we get the best accuracy. We can evaluate the result how many wrong chunks are tagged and how many chunks can be correctly tagged. The grammar-based systems have limitations because natural language often does not conform to the rules of the grammar. Unusual constructions, casual speech, innovative expressions, mistakes, noise, and interruptions can all result in sentences that are quite understandable to a human reader or listener, but utterly confusing to a rule-based system. It is hard to write a complete and tight grammar.

Therefore, the performance of our chunker is evaluated in terms of problems that can be encountered in Myanmar sentences because of some peculiar patterns. The sentences that have peculiar patterns are entered into the system and check the accuracy of our chunker. In this system, there are two portions: normalization and chunking so that evaluation must be performed on normalization at first. Some errors can occur in the normalization step for some words especially for negative word because unusual pattern of verbal negation is found in such patterns where the second verb of a compound is marked with the negative prefix, as in ne ma-kaung: နေမ-ကောင်း: 'unwell', nar ma-lal: နားမ-လည် : 'misunderstand', etc. To alleviate these errors, we have to be inserted these words in the POS tagged corpus before normalization and their tags should be basic verb tags at that time.

For chunking, our evaluation result is depended upon normalization result. If normalization builds the correct finer tags, chunker will determine the right chunks. In the chunking step, error can occur when too many noun tags or verb tags are appeared continuously, chunker will assemble them in only one chunk and identify as one noun chunk or one verb chunk. It can be encountered in compound verb or noun.

We have evaluated our rules in terms of the number of correct chunks it can recognize i.e recall. We have achieved a high recall of 97.5% on the POS tagged corpus with normal pattern text and 92.06% on corpus including peculiar pattern sentences.

8. Conclusion

This paper proposes an implementation of chunk tagged corpus using rule based approach. Lexical rules have to be applied to normalize some words and tags in order to produce accurate and finer tags. For the input, a Myanmar POS tagged corpus, which is developed by [8], has to be used. The annotation standards for chunk tagging include 7 tags. "Myanmar-English Dictionary" [6] and "Myanmar Grammar" [7] books published by Myanmar Language Commission are used as references for POS tagging and chunking of Myanmar words. One of the improvements to be done is adding more lexical rules in order to do more accurate normalization. Another is adding more chunking rules to get better performance.

For future work, we hope to conduct more experiments to examine how different types of input affect the performance. This chunk tagged corpus can be used in a number of NLP applications. In Myanmar to English machine translation system, Grammatical Function Assignment, Word Sense Disambiguation, Translation Model and Reordering systems have to use these chunk tags for analyzing Myanmar words in order to translate Myanmar text to English text.

References

[1] Abney, S. P., "Parsing by Chunks", *Principle-based parsing: computation and psycholinguistics edition*, Kluwer Academic Publishers, Dordrecht, 1991.

[2] Abney, S. P., "Partial Parsing via Finite-State Cascades", *In Workshop on Robust Parsing, 8th European Summer School in Logic, Language and Information*, pages 8–15, Prague, Czech Republic, 1996.

[3] Boehm, I., "Rule based vs. statistical chunking of CoNLL data sets", 2005.

[4] Hopple, P. M., "*The structure of nominalization in burmese*", Ph.D Dissertation, University of Texas, Arlington, 2003.

[5] Kumar, G. B. and K. N. Murthy, "UCSG: A Wide Coverage Shallow Parsing System", *In proceedings of IJCNLP*, 7-12 January 2008.

[6] "*Myanmar-English Dictionary*", Department of the Myanmar Language Commission, Ministry of Education, Myanmar, 2006.

[7] "*Myanmar Grammar*", Department of the Myanmar Language Commission, Ministry of Education, Myanmar, 2005.

[8] Myint , P.H., "Assigning automatically Part-of-Speech tags to build tagged corpus for Myanmar language", *The Fifth Conference on Parallel Soft Computing*, Yangon, Myanmar, 2010.

[9] Tjong Kim Sang, E. F. and S. Buchholz, "Introduction to the CoNLL-2000 shared task: Chunking", *Proceedings of CoNLL-2000 and LLL-2000*, Lisbon, Portugal, pages 127-132, 2000.

[10] Vuckovic, K., M. Tadic and Z. Dovedan, "Rule based chunker for Croatian", *In Proceedings of LREC*, 2008.

Appendix

Table 1.

Customized chunk tagset

No.	Tag Name	Description	Example
1.	NC	Noun Chunk	ကလေးများ၊ သူတို့၏စာအုပ်၊ ကျွဲအိမ်
2.	JC	Adjectival Chunk	လှ၊ လိမ္မာ
3.	RC	Adverbial Chunk	မနေ့က၊ ဂရုစိုက်၍၊
4.	COC	Conjunctional Chunk	သို့သော် ၊ နှင့်
5.	PPC	Prepositional Chunk	သို့ ၊အတွက် ၊အတိုင်း
6.	VC	Verb Chunk	ကျွေးဖူးတင်၊ ပြုလုပ် ၊ ပြောကြား
7.	SFC	Sentence Final Chunk	သည်၊ ပါ ၊ ပါသည်၊

Table 2.

Customized basic POS tagset

No.	Tag Name	Description
1.	NN	Singular Noun
2.	NNP	Proper Noun
3.	PRN	Pronoun
4.	JJ	Adjective
5.	RB	Adverb
6.	VB	Verb
7.	CC	Conjunction
8.	PART	Particle
9.	PPM	Postpositional Marker
10.	INJ	Interjection
11.	CRD	Cardinal Number
12.	ORD	Ordinal Number
13.	SYM	Symbols
14.	SF	Sentence Final

Table 3.

Customized finer POS tagset

No.	Tag Name	Description
1.	NNR	Plural Noun
2.	JJC	Comparative Adjective
3.	JJS	Superlative Adjective
4.	RBC	Comparative Adverb
5.	RBS	Superlative Adverb
6.	NEG	Negative