

Doctoral Dissertation

**Chinese Synthetic Word Analysis using
Large-scale N-grams and
An Extendable Lexicon Management System**

Jia Lu

February 20, 2011

Department of Information Processing
Graduate School of Information Science
Nara Institute of Science and Technology

A Doctoral Dissertation
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
Doctor of ENGINEERING

Jia Lu

Thesis Committee:

Professor Yuji Matsumoto	(Supervisor)
Professor Kiyohiro Shikano	(Co-supervisor)
Associate Professor Masashi Shimbo	(Member)
Assistant Professor Masayuki Asahara	(Member)

Chinese Synthetic Word Analysis using Large-scale N-grams and An Extendable Lexicon Management System*

Jia Lu

Abstract

The lack of internal information of words has become a crucial problem for morphological analysis systems, especially for languages like Chinese and Japanese, in which long un-separated characters are often used as one single unit. This problem gets even clear when one apply the segmented result of sentence to upper NLP applications such as machine translation. If there is no information of the internal structure of long words, it is difficult to match the correct meaning between two languages' word pair.

In this dissertation, we first define the concept of synthetic word and describe differences between single-morpheme words and synthetic words within Chinese language. Then we discuss the categorization of Chinese synthetic words according to their internal syntactic relation or morphological structure on linguistics base. Next we propose several approaches to parse the internal structure of synthetic words and conduct experiments for each of these approaches. After trying to parse synthetic words based on general adjacency model and dependency model, we use large-scale n-gram data and customized CYK algorithm for analysis. In detail, we use SVM to generate margin score for each character sequence, which could be a possible internal part of the target synthetic word. Then we consider these margin scores as weights to construct a best tree structure for that synthetic word using a customized CYK parsing algorithm. Finally, we

*Doctoral Dissertation, Department of Information Processing, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DD0861027, February 20, 2011.

will give experiment results on parsing synthetic words, investigate these results from several aspects and indicate how further improvement could be made.

At the end of this dissertation, we describe an extendable lexicon management system that we created for easy word internal structure annotation. After explaining the motivation and reason of system development, we show how to use it to manage normal lexicon information and annotate internal structures of synthetic words. Then based on the user experiences and feature requests until now, we describe some shortages of current system. And we also propose several possible improvements that could be made in future in order to let this system be much more user-friendly and powerful for managing all kinds of lexical resources.

Keywords:

synthetic word, morpheme, internal structure, classification, parsing, annotation

Acknowledgements

First of all, I want to thank Prof. Matsumoto who has always been very kind to me after I came to Japan. It would be impossible for me to settle down in Naist and start my research smoothly without professor's help. And as supervisor, he always gives me insightful advices that lead me to the right way of my research.

I would also like to thank Assistant Professor Aasahara for his continuous discussions and advices on my research, which are indispensable for me to make progress on my work. Especially in the time of my Ph.D course, every time when I felt lost or disappointed on my research, his warm words always gave me strength to stand up again and move forward. If it would not be his sincerely help, I would not be able to finish doctor dissertation and write this acknowledgements. And I also want to thank him for his help in my private life for giving me really useful advices about raising a baby in Japan.

I would also like to thank Professor Shikano and Associate Professor Shimbo for the great patience on reading this thesis and the valuable comments on it.

I also want to thank Chooi-Ling Goh and Yuchang Cheng, who gave me really helpful advices when I first came to Naist without any knowledge of natural language processing. Without their advice and encouragement, I may not be able to start my research quickly and finish this research until now.

Finally, I want to thank all my colleagues in Computational Linguistics Laboratory. I appreciate all their kindness in helping me settling down in Japan, starting my research and solving problems encountered during these years. I want to especially thank Assistant Professor Komachi, Manabu Kimura and Yotaro Watanabe for their valuable advices both on my research and my private life in Japan.

To my family, I would like to thank my parents for teaching me the value of education and kindness at a young age. And I also want to thank them for always trusting me and supporting me to study abroad. To my wife and my wonderful daughter, I appreciate their encouragement and support, so that I can complete my study. And thank them for their wonderful smile everyday that has given me the strength to finish all these hard going works.

Contents

Acknowledgements	iii
1 Introduction	1
1. Background	1
2. Two issues in Chinese word segmentation	2
3. Objectives and benefits and of this research	5
4. Dissertation outline	7
2 Chinese Synthetic Word	9
1. Definition of word in Chinese	9
2. Classification of Chinese synthetic words	12
3. Morphologically derived words	14
4. Compound words	18
5. Exceptions	20
6. Research targets	21
3 Related Work on Synthetic Word Analysis	23
1. A Chinese synthetic word corpus	23
2. Rule-based approach on synthetic words	24
3. Chinese multiword chunking	25
4. Analysis on Japanese medical terminology	27
5. Structural analysis on English noun phrase	27
6. Other related researches	28
4 Parsing Three-character words with Adjacency Model	29
1. Two basic parsing models for three-character synthetic words	29

2.	Statistical models and tools	30
2.1	Supervised and unsupervised machine learning method . .	30
2.2	Mutual information	31
2.3	Support vector machines	31
3.	Dataset	32
4.	Experiment	36
4.1	Experiment one: using mutual information only	36
4.2	Experiment two: using SVM classifier	37
4.3	Experiment three: classification on compound words . . .	39
5.	Summary	41
5	Parsing Three-character Words with Dependency Model	43
1.	Analysis using dependency parsing approach	43
2.	MST Parser	44
3.	Experiment	45
3.1	Transformation from dependency tree to structure tree . .	45
3.2	Dataset and feature	46
3.3	Experiment result and discussion	47
4.	Summary	50
6	Tree-based Parsing Approach for All Synthetic Words	51
1.	Tree-based parsing approach	51
2.	Top-down approach vs. bottom-up approach	54
3.	Parsing algorithm	56
3.1	Original CYK parsing algorithm	56
3.2	Customized CYK parsing algorithm	58
4.	Structure annotation of synthetic words	61
5.	Experiment	62
5.1	Detailed parsing method	62
5.2	Features preparation	64
5.3	Experiments on three-character synthetic words	69
5.4	Experiments on all synthetic words	70
6.	Summary	73

7	An Extendable Lexicon Management System	75
1.	Background	75
2.	Aims and tools	76
3.	System Design and functionality	78
4.	Implementation and remaining problems	85
8	Conclusion and Future Work	87
1.	Conclusion	87
2.	Future work	88
	References	91
	Appendix	97
	A. POS tagset in ChaSen Chinese Dictionary	97
	List of Publications	99

List of Figures

1.1	Tree structure and system output for ordinary word	6
1.2	Tree structure and system output for merging pattern	6
2.1	Relationship of word, morpheme and character in Chinese	9
2.2	Relationship among Chinese word, synthetic word, MDW and CW	13
2.3	Types of morphologically derived words	14
3.1	Synthetic word tagging approach	24
3.2	Dependency parsing on Japanese Medical Terminology.	27
4.1	Sample of Chinese GigaWord	33
5.1	Transformation from dependency to tree structure	44
5.2	Transformation from dependency to tree structure using edge labels	46
5.3	Three-character dependency trees	47
5.4	Three-character word dataset for MSTParser	47
5.5	All possible combinations considered by MSTParser	49
6.1	An example of parse tree of long synthetic word	52
6.2	Different parsing tree with same POS sequence	53
6.3	Top-down approach and bottom-up approach for analyzing long synthetic words.	54
6.4	CYK construction table for a synthetic word.	57
6.5	Prototype of customized CYK algorithm	60
7.1	An example of lexicon information expansion	75
7.2	Database design of Cradle	78
7.3	Search interface of Cradle	80

7.4	Search result of Cradle	81
7.5	Tree structure annotation interface of synthetic word in Cradle . .	83
7.6	Tree structure view of a synthetic word in Cradle	84
8.1	New system design of Cradle	90

List of Tables

1.1	Example of segmentation patterns with different granularity . . .	5
2.1	Overlapping of morphologically derived words and compound words	13
2.2	Different patterns of reduplication in morphologically derived words	15
2.3	Different patterns of merging in morphologically derived words . .	16
3.1	Types of multiword chunk in Chinese	25
4.1	Distribution of words more than two characters	34
4.2	Distribution of MDW words' categories in 1,000 words	35
4.3	Distribution of compound words' categories in 1,000 words	35
4.4	Number of words in reduplication structure	35
4.5	Features for classification on morphologically derived words	38
4.6	Results of SVM on classfying left and right branching internal structure	38
4.7	Results after adding more features by referencing CGW	39
4.8	Features for classification on compound words	40
4.9	POS re-annotation rules for compound words	40
4.10	Results after part-of-speech modification	41
5.1	Label set for transforming from dependency tree to structure tree	45
5.2	Experiment results of MSTParser on three-character synthetic word	48
6.1	Internal part candidates for a special word	53
6.2	Internal part candidates for a special word	58
6.3	Complete internal structure candidates for character sequence ‘激 光打’	59

6.4	Candidate tree numbers generated by CYK when doing complete search	60
6.5	Distribution of annotated synthetic words	62
6.6	Possible internal candidates for 副国防部长	63
6.7	Example of suffix array for character sequence ‘abracadabra’ . . .	65
6.8	PMI feature selection for SVMs	68
6.9	Distribution of annotated three-character synthetic words	70
6.10	Classification and parsing results of three-character synthetic words	70
6.11	Classification and parsing accuracy for all synthetic words	71
6.12	Classification performance with different internal parts’ length . .	71

Chapter 1

Introduction

1. Background

Over the past decade, because of the rapid expansion of Internet, a large amount of corpus, either generated from the web or hand-annotated, became available to researchers in the field of Natural Language Processing (NLP). This gives a huge push for approaches taken for NLP tasks to shift from rule-based pattern to statistical pattern, which consequently leads to big performance improvements in a lot of areas of natural language processing. This is the reason why nowadays the end users can enjoy those practical services like search engine, voice recognition and automatic machine translation, which are supported by various kinds of NLP technologies.

Another good aspect of this rapid NLP technology emergence is its availability for a wide range of natural languages, from major language like English or French to minor language like Vietnamese etc. Chinese, a language spoken by the largest population around world, has also gained benefits from these NLP technologies.

As it is widely known, Chinese language does not use spaces to indicate word boundaries as English and other Latin languages do. Therefore, though there are many other features in Chinese besides the problem of no delimitation, such as no inflection, word segmentation is believed as the very prior and fundamental step in Chinese natural language processing. Furthermore, because Chinese has only one type of character (Hanzi) as compared to other un-separated languages like Japanese, which has three types of characters (Kanji, Hiragana, Katakana), it

makes the task for delimiting word boundaries in Chinese much more difficult than in other un-separated languages. Actually, even for native Chinese speakers, while there is an agreement on that a string of Chinese characters could be segmented into words, there are a substantial number of cases where no agreement can be reached on how to segment a sequence of characters and to what granularity the segmentation process should be done.

Because word segmentation is the first step in Chinese natural language processing, its output, sequence of segmented words, is used by almost all kinds of upper NLP applications. Therefore, it is considered that the result of morphological analysis system, which produces the final sequence of segmented words, should satisfy all upper applications' needs in a generalized manner. In recent years, along with the whole progress of NLP community, several institutes around the world have focused their attention on this difficult and challenging task: Chinese word segmentation. However, though they reported good performance on their system, none of their works can satisfy all kinds of needs with a generalized purpose. They are either self-domain specific or having performance drop down on some particular problems.

Commonly, it is believed that these problems come out because of lacking internal information of Chinese words. In fact, the knowledge of word internal information represents the construction process of Chinese word. Native Chinese speakers can recognize words from a sequence of characters because the pattern of Chinese word construction is hiding in their brain. Although the knowledge of Chinese word construction pattern is debated heavily among linguists, the researchers of NLP field have not paid much attention on this topic. In this dissertation, we will focus on how to recognize internal word information of Chinese using statistical NLP methods, and we believe it is necessary to apply this kind of information to morphological analysis system in order to fit various upper NLP applications.

2. Two issues in Chinese word segmentation

Although Chinese word segmentation is a difficult and challenging task in Chinese language processing, at the time of writing, several Chinese morphological analy-

sis systems were developed by various institutions after trying a lot of algorithms and statistical models based on either words or characters, and they all achieved quite good performance on Chinese word segmentation [6, 8, 26, 21, 1, 7, 38, 33]. However, there still remain two crucial issues in Chinese word segmentation process, which slow down the progress of Chinese NLP research and need attention.

- **No unified segmentation standard**

The biggest problem is that though most of the Chinese morphological systems have quite good performance, none of them can satisfy all levels of needs of upper natural language processing applications.

The main reason is that each morphological analysis system has its own segmentation standard and language resources, which means there is no single segmentation standard for all tagged corpora that can be agreeable across different research groups. As a result, most of these systems have their best performance only on their own target resources, and cannot generate various levels of segmentation patterns other than the one fitting their own segmentation standard.

However, while more and more NLP-based applications or services are created, they will in turn have different requirements that call for different granularities of word segmentation as their basic input. For instance, a long (upper level) segmentation unit like “洗衣粉 (detergent)” may simplify syntactic analysis and application like IME, but small (lower level) segmentation units like “洗衣 (washing)” and “粉 (powder)” might be better for information retrieval or word based statistical summarization.

Even for a single application, such as machine translation (MT), the knowledge of both lower and upper level segmentation units will be helpful to improve system performance. For example, the knowledge that “烤面包器 (toaster)” can be divided into “[烤 (toast), 面包 (bread)], 器 (device)” will be a good clue for MT system to translate “烤面包器” to “toaster” correctly.

Unfortunately, present Chinese morphological analysis systems can only generate segmentations in one level, which differs according to various standards. So from the viewpoint of practical application, it is necessary to find a

way to make segmentation units more flexible in morphological analysis process.

Among all differences of segmentation standards, segmentation pattern for Chinese synthetic words is the most controversial part because Chinese synthetic words have the most complex structures and should be represented by several segmentation levels according to the needs of upper applications. Therefore it is necessary to first analyze the internal structure of Chinese synthetic words before improving the whole performance of morphological analysis systems.

- **Low efficiency of out-of-vocabulary word detection**

Another crucial problem in recent morphological analysis systems is that the detection of out-of-vocabulary words is still a difficult task until now. It is mainly because only looking at the context of this kind of words cannot easily recognize them.

Intuitively, to eliminate of out-of-vocabulary words, one could manually add more and more words into system dictionary so that the ratio of out-of-vocabulary words can be reduced in actual segmentation procedure. But in fact, as for any other languages, even the largest dictionary we may create, will not be able to register all possible words such as proper names, numbers, and etc. This is particularly true for Chinese because almost any character can be used to form new words. Therefore, a proper solution to detect the unknown words is necessary.

Present methods of OOV word detection, such as [9, 10], make use of machine learning methods to automatically extract out-of-vocabulary words from real text during segmentation and register them to dictionary of the morphological analysis system. However, comparing to Japanese, in which new words, especially foreign words, usually takes the form of katakana characters according to their pronunciation, most native Chinese speakers intend to make new words or translate words from foreign languages with characters according to their meanings. This kind of flexibility of Chinese word construction makes the present OOV word detection quite inefficient and there are still many out-of-vocabulary words that could not be easily recognized.

Based on the work [10] on out-of-vocabulary word detection and annotation, we find that most out-of-vocabulary words are proper nouns like organization names, technical terms, etc. Because these kinds of proper nouns usually take the form of Chinese synthetic words, we believe that Chinese synthetic word analysis could help us on the detection of out-of-vocabulary words, which will finally improve the whole performance of morphological analysis.

3. Objectives and benefits and of this research

Objectives

In this research, we use machine-learning methods to parse Chinese synthetic words into several internal parts at every granularity level. Then we produce tree structures of target synthetic words by using this information and finally store these structures into a synthetic word dictionary.

The following Table 1.1 shows the most ordinary kind of words in Chinese, which have prefix or suffix constituents.

Table 1.1. Example of segmentation patterns with different granularity

副国务院发言人	vice spokesman of state council
	副、国务、院、发言、人
副国务院发言人	副、国务院、发言人
	副、国务院发言人

As shown in Table 1.1, this kind of words can be segmented into several different levels according to their internal syntactic relation. However it is quite difficult for one to determine how deep it is the best to go in segmentation process. In our work, we want to provide the whole structure information for every synthetic word, so we represent all levels of structure information using a tree, as shown in Figure 1.1.

Besides the above ordinary tree structure, there is a special kind of phenomenon in Chinese word formation process, where two words create a truncated

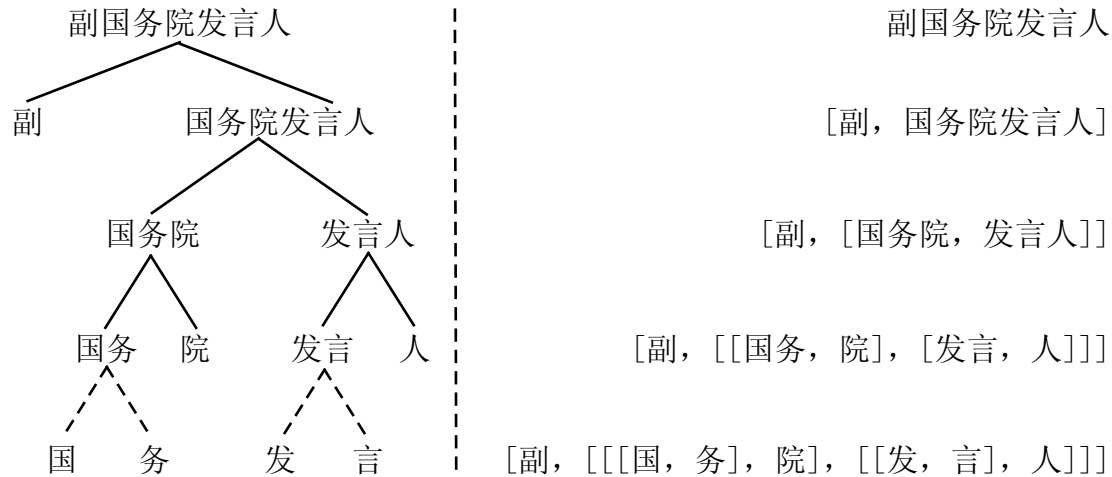


Figure 1.1. Tree structure and system output for ordinary word

(or merged) form using common internal part when they are concatenated. In this case, we cannot describe their segmentation pattern using ordinary approach (a string with parenthesis indicating levels and parts). But we can illustrate it using a tree structure. The tree structure and system output for this kind of words are shown in Figure 1.2. And the concept of this kind of words will be further discussed in Chapter 2.

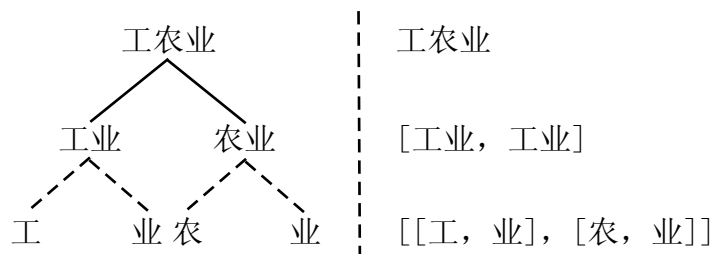


Figure 1.2. Tree structure and system output for merging pattern

The final objective of our work is to use machine-learning methods to correctly parse Chinese synthetic words into structure trees and store these tree information into a well-formed synthetic word dictionary.

Benefits

In order to improve the whole performance of present morphological analysis system, we need to generate flexible segmentation output for different kinds of upper NLP applications and recognize more out-of-vocabulary words. This in turn requires us to analyze Chinese synthetic words for the knowledge of internal word information. As the following shows, we believe that there are several benefits for doing this research:

- By parsing every Chinese synthetic word into a possible tree structure, it will let Chinese word segmentation system give a more reasonable output for different kinds of upper application with various granularities.
- It will make the detection of out-of-vocabulary words much smoother by taking advantage of the internal knowledge of Chinese synthetic words.
- It will make existing dictionary of morphological analysis system much more flexible by linking words into tree structures. And this resource will be very useful for analysis word relation in syntactic level.

4. Dissertation outline

The remainder of this dissertation is organized as follows: Chapter 2 introduces the concept and categorization of Chinese synthetic words on linguistics base. Chapter 3 describes several previous work related to synthetic words. Chapter 4 first describes two models (adjacency model and dependency model) usually used on parsing synthetic words. Then it shows our approach to analyze three-character synthetic word with golden-standard classification method based on adjacency model. Next, Chapter 5 describes the parsing experiments on three-character synthetic word analysis based on dependency model. Chapter 6 explains our analyzing approach using classification and customized CYK parsing algorithm for all synthetic words. Chapter 7 introduces an extendable lexicon management system we created, which can store tree information of synthetic words. We show how this system can ease the annotation work of synthetic words' structure and we describe some ongoing development and plans to make

this system better. Chapter 8 concludes this dissertation and mentions future work.

Chapter 2

Chinese Synthetic Word

1. Definition of word in Chinese

There has always been a common belief that in Chinese a ‘word’ is by no means a clear and intuitive notion and Chinese ‘does not have words’, but instead has ‘characters’, or that Chinese ‘has no morphology’ and so is ‘morphologically impoverished’. But actually for native Chinese speakers, words are lexical entries, representing a complete concept and occurring innately in the form of specific language rules based on the speaker’s mental lexicon. Figure 2.1 shows the basic relationship of word, morpheme and character in Chinese.

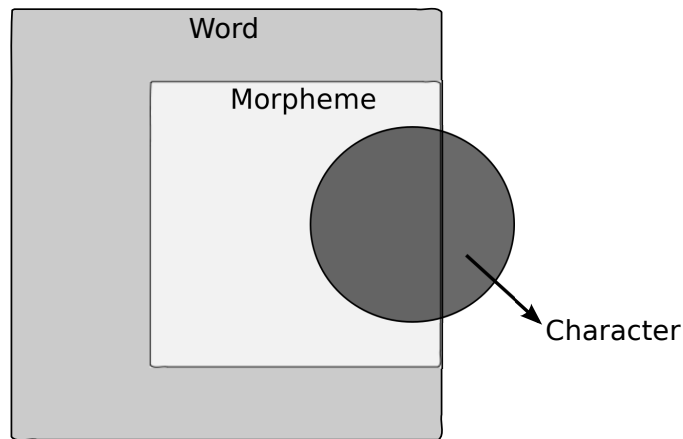


Figure 2.1. Relationship of word, morpheme and character in Chinese

However, while native Chinese speakers are often able to agree on how to segment a string of characters into words, there are a substantial number of cases where no agreement can be reached. Until now, although there are a lot of ways to classify Chinese words from the linguistics aspect, there is no word segmentation standard widely agreed in Chinese according to [4] on Chinese grammar.

In the field of pure linguistic, most researchers focused their attention on the formation rules of two-character words, the most ordinary words in Chinese, which also have the most flexible formation structure. Some of them divide words into categories based on their part-of-speeches and internal syntactic relations. Some of them classify words into ‘content words’ and ‘empty words’ based on whether those words have actual meanings or not. There are also some researchers who analyze the internal structure of words using the concepts of ‘free morpheme’ and ‘bound morpheme’ according to [29].

However, from the computational linguistics point of view, our main purpose in word segmentation is to get correct words with correct part-of-speech from text, therefore it is useless for us to break every Chinese word into single characters and analyze their internal relation in the segmentation process. Furthermore, because Chinese native speakers often use two-character words as single words literally and most of two-character words have already been registered as lexical entries in our Chinese dictionary, our main targets in Chinese synthetic word analysis are those words having more than two characters. But before that, we need to first understand what synthetic word is. Based on [4], we believe that it is proper to first divide Chinese words into the following two kinds according to the way their internal parts constructed.

- Single-morpheme words
- Synthetic words

Single-morpheme words are those words that only have one morpheme inside and cannot be divided into smaller parts when representing as a whole concept. It means if we divide single-morpheme words into characters or parts, the meanings of individual parts are independent with each other and do not indicate the meaning of the original word. More specifically,

single-morpheme words have the following three different types, and should be segmented as one word in any morphological analysis systems.

① one-character words

Apparently, one-character words are single-morpheme words, because characters are the smallest units in Chinese and they cannot be further divided. An ordinary character in Chinese stands for an independent morpheme with one or several senses. Such as:

人 (human), 马 (horse), 车 (vehicle)

② one-morpheme words

This kind of words belongs to single-morpheme words though they have more than one character inside. It is because in Chinese, the internal characters of these words do not have any particular meanings if separated. And they can only become an actual morpheme when bound together to form into word. So this kind of words can only act as one morpheme and cannot be further divided. Such as:

A: 鹌鹑 (quail), 翡翠 (jadeite), 鸳鸯 (mandarin duck)

B: 蝈蝈 (grasshopper), 猩猩 (orangutan), 狒狒 (papio)

C: 咔嚓, 哗啦, 滴答 (onomatopoeias, indicating some specific sound)

③ transliteration words

There are a lot of transliteration words in Chinese, which use Chinese characters to simulate the pronunciations of foreign words. Word of this kind cannot be broken into parts because the separated parts do not have any connection with the original meaning. Examples of transliteration words are as follows:

比萨 (pizza), 肯德基 (Kentucky), 阿司匹林 (aspirin)

As explained above, if we divide 肯德基 (Kentucky) into ‘肯 (can)’, ‘德 (moral)’ and ‘基 (base)’, it definitely cannot indicate the meaning of the well-known fried chicken restaurant chain from those three characters.

Synthetic words are those words that are composed of two or more single-morpheme words and represent a new entity or meaning which can be in-

licated from the internal constituents. According to this definition, synthetic words are more complicated than single-morpheme words, because if we divide synthetic words into smaller parts, we can still somehow guess the meaning of the original word from the meanings of the internal parts despite the fact that it may not be a very precise one.

For example, if we do not know the meaning of ‘司机 (driver)’, but we do know the meaning of ‘司’ is ‘control’ and the meaning of ‘机’ is ‘machine’. Then we can guess that the meaning of ‘司机’ may be connected with ‘control’ and ‘machine’, and actually the real meaning is the person who drives (controls) a car (machine).

2. Classification of Chinese synthetic words

The synthetic words may be understood as the result or output of word formation rules in Chinese language. Classification of these Chinese synthetic words is a difficult task because the formation rules are not so obvious and sometimes even a native speaker cannot determine which category a word should belong to. Actually in pure Chinese linguistics theories, categorization of synthetic words is still quite controversial, and sometimes even the terminologies themselves are not having a unified standard.

However, when native Chinese speakers analyze a synthetic word, they intend to first divide the word into parts before determining the internal syntactic relation of it. Therefore in our work, we first define Chinese synthetic words from two points of view, which are internal morphological structure of word and internal syntactic relation of word.

From the internal morphological structure point of view, we call synthetic words as morphologically derived words, and from the internal syntactic relation point of view, we call synthetic words as compound words.

- Morphologically derived words
- Compound words

Note that these two notions are not exclusive of each other. Actually they overlap with each other in most cases. In other words, a Chinese synthetic word

is usually called a morphologically derived word when people emphasize on its morphological forming structure. While it is called a compound word when people pay their attention to the inside syntactic relation between internal constituents. Figure 2.2 shows the relationship among these concepts.

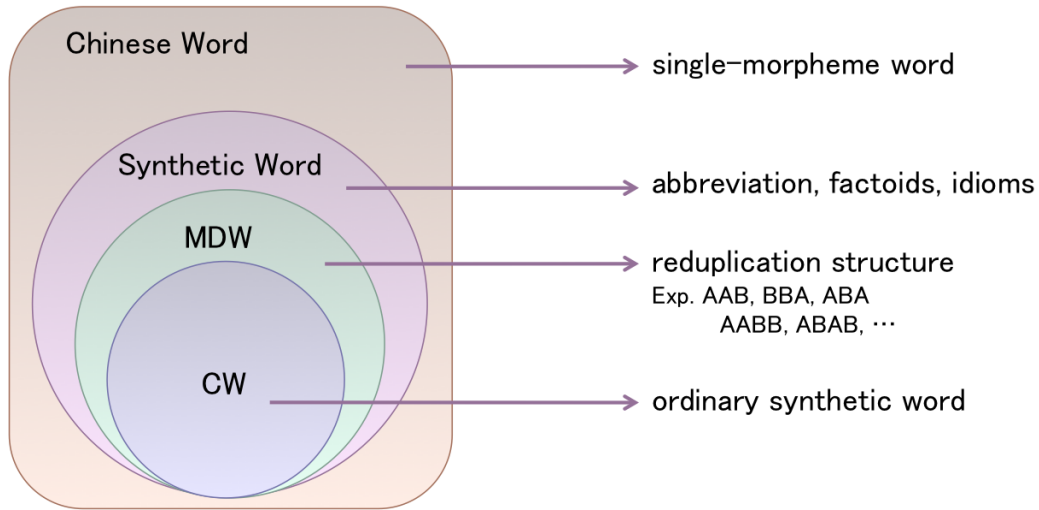


Figure 2.2. Relationship among Chinese word, synthetic word, MDW and CW

Table 2.1. Overlapping of morphologically derived words and compound words

Morphologically derived word	Compound word
Left branching (AB/C)	Head modifier (M-H)
面包 / 店	面包 - 店
服装 / 店	汽车 - 站
咖啡 / 店	电话 - 亭

For example, as shown in Table 2.1, ‘面包店 (bakery)’ can be considered as a morphologically derived word because it has a left-branching structure pattern ‘XX / 店’, which is the same like ‘服装店 (clothes store)’ and ‘咖啡店 (coffee shop)’. Meanwhile, it is also a compound word because the relation between its internal parts ‘面包 (bread)’ and ‘店 (shop)’ is head-modification, which is the same like ‘汽车站 (bus stop)’ and ‘电话亭 (telephone booth)’.

In the following sections, we introduce two sorts of categories: one for morphologically derived words and the other for compound words. The differences among these categories will become clear after analyzing the given examples. Here we will only give several examples for each category, more detailed conceptual definitions can be found in our previous work.

3. Morphologically derived words

Morphologically derived words have specific morphological internal structure. They are called morphologically derived words, because in Chinese if one knows the morphological internal structure of a particular word, it is easy to create a lot of similar words by using the same construction pattern. Because this kind of words, which are very dynamic and productive, are usually the results of morphological construction process, they are the most controversial ones and thus are most likely to be treated differently in different standards or by different morphological analysis systems.

Although there are a lot of different opinions on the classification of morphologically derived words, however, from the computational linguistic aspect, we believe that morphologically derived words should be categorized into three types shown in Figure 2.3 based on the discussion in Packard's book [29].

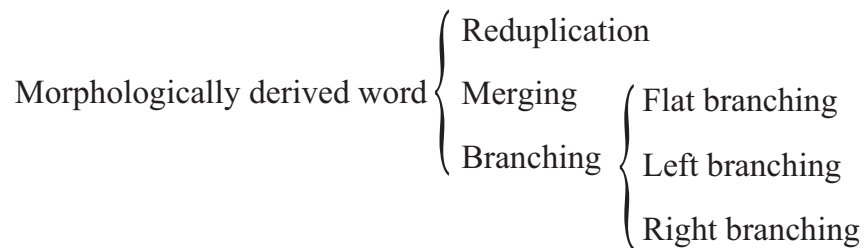


Figure 2.3. Types of morphologically derived words

- **Reduplication**

Reduplication is a language phenomenon that is peculiar in Chinese. It means

that characters reduplicate themselves inside words, which can often make the original meanings of words emphasized or make the the accent of words softer.

Usually this phenomenon appears in words with less than five characters and there are mainly eight patterns of reduplication in Chinese. They are AA, AXA, AAB, ABB, ABAB, AABB, AXAY and XAYA, where A, B, X and Y stand for distinct characters. The following Table 2.2 shows examples of these patterns.

Table 2.2. Different patterns of reduplication in morphologically derived words

AA	听听 (listen)	AXA	看了看 (watched)
AAB	帮帮忙 (help)	ABB	雄赳赳 (valiantly)
ABAB	研究研究 (research)	AABB	高高兴兴 (happy)
AXAY	跑来跑去 (run around)	XAYA	东看西看 (look here and there)

As shown in the examples, these words can all be considered as single words because breaking them up results in segments that are not independent words. But this is not true for these three types: AA, AXA and ABAB, because breaking up AA and AXA results in one-character words A and X while breaking up ABAB results in independent word AB. However, according to [34], these three types should be also considered as synthetic words in that they will serve different levels of segmentation need of natural language processing applications.

• Merging

Merging is a language phenomenon that is quite ordinary in both Chinese and Japanese. It means two semantically related words, which have an internal part in common, can merge into one word by removing one of the common parts.

This morphological phenomena, also known as ‘telescopic compounding’ in [15], can be considered as a sub-case of abbreviation. But unlike other kinds of abbreviation, it has fixed patterns and a predictable semantic interpretation. Usually, there are three merging patterns shown in Table 2.3.

Table 2.3. Different patterns of merging in morphologically derived words

①	Left merging: $AB + AC \rightarrow ABC$ Example: <u>国</u> 内 + <u>国</u> 外 \rightarrow <u>国</u> 内外 (domestic and abroad)
②	Right merging: $AC + BC \rightarrow ABC$ Example: 中 <u>学</u> + 小 <u>学</u> \rightarrow 中小 <u>学</u> (middle school and primary school)
③	Middle merging: $AB + BC \rightarrow ABC$ Example: 北京 <u>市</u> + <u>市</u> 长 \rightarrow 北京 <u>市</u> 长 (Mayor of Beijing city)

As shown in Table 2.3, these patterns have some differences between each other in the meanings of whole words. We can find this difference when we examine the English translation of the above examples carefully: the first two types have an ‘and’ in their translation while the last type has an ‘of’ in it. It is because that in type ① and ②, the two meanings of internal parts are parallel, thus the meaning of whole word is just the sum of the two. While in the case of type ③, one internal part is the head and the other is the modifier, which in the end make the meaning of whole word has a head-modification relation inside.

Some may argue that the first two types can be seen as A/BC (国/内外) and AB/C (中小/学), in which the longer internal part can be seen as a valid word having parallel internal relationship, like 内外 (inside and outside) and 中小 (middle and small). Based on more than 10,000 words we examined, we found that words having left merging structure can always be seen as A/BC, where BC is a valid word and has a parallel internal relationship. However, that is not always the case for words having right merging structure. For example, 动植物 (animals and plants) or 错别字 (wrongly written or mispronounced characters) can not be seen as AB/C, because AB (动植 or 错别) is not a valid word though it may have parallel relationship between A and B.

However, in actual bottom-up parsing process, though 动植 and 错别 are not valid words, they still can be generated by a CFG parser. In other words, basically there are two way to let a parser know where a merging action happens:

- In structure layer. Using the pattern shown in Table 2.3, where merging actions are represented by special construction patterns.
- In syntactic layer. Using normal CFG construction patterns for left or right merging, where the meaning of merging is represented by those labels annotated on internal parts of parent word.

For simplicity, we use the first way to consider all merging structure by using special construction patterns, described in Table 2.3.

• Branching

Branching is a very productive morphological process in Chinese. Almost all words in Chinese have branching internal structure. And the information of these branching structures is quite useful for native speakers to determine the internal syntactic relations of words. Here, we divide branching into flat branching, left branching and right branching based on their number of parts or characters on each segmenting level.

Flat branching There are two cases when we can say a synthetic word has flat branching structure. One case is if the number of word internal parts on the same level is bigger than two, then we say the word's structure is flat branching. The other one is if one word has only two parts and both of them have same character length, we say the word's structure is flat branching too.

When a word with flat branching internal structure only has three characters, the character in the middle is possible to be an infix character. The following are some examples with flat branching internal structure. The underline shows the infix.

- two internal parts: 冷嘲/热讽 (ironicalness)
- four internal parts: 东/南/西/北 (east, south, west, north)
- three internal parts without infix: 中/日/韩 (CJK)
- three internal parts with infix: 看/不/到 (cannot see), 听/得/见 (can hear)

Left branching Internal structure is a binary combination of two parts, in which the length of the left part is longer than the length of the right part. Usually the right part is a single morpheme word or a two-character synthetic word, which functions as the head part of the whole word.

If the whole word is a three-character word, then left branching internal structure is very possible to be a typical suffix structure in which the right part is a suffix character frequently used in Chinese. The following are some examples with left branching internal structure. The underline shows the suffix.

- word having more than three characters: 无神论/者(atheist)
- three-character word: 洛阳/市(Luoyang city), 安全/厅(security agency)

Right branching Internal structure is also a binary combination of two parts like left branching. But this time the length of the left part is shorter than that of the right part. And the left part is usually a single morpheme word or a two-character synthetic word, which functions as the modifier part of the whole word.

If the whole word is a three-character word, the right branching internal structure is very possible to be a typical prefix structure in which the left part is a prefix character frequently used in Chinese. The following are some examples with right branching internal structure. The underline shows the prefix.

- word having more than three characters: 总/工程师 (Executive Engineer)
- three-character word: 副/主席 (vice president)

4. Compound words

Compound words are words whose internal constituents have a certain kind of syntactic or semantic relations with each other. Though this concept of compound words is quite simple, it is actually a difficult task to analyzing these words.

In Chinese, the number of compound words is the largest because almost all Chinese words are created based on the internal syntactic rules according

to the meanings and functions of different constituents. Also, recognizing these internal syntactic relations is especially difficult for computers. It is because sometimes those inside syntactic relations of compound words are so ambiguous that even native speakers cannot confidently determine which syntactic relations they should belong to.

Here, based on [13]’s discussion on Chinese synthetic words, we believe that compound words can be divided into the following five categories.

- **Subject-predicate**

Words of this kind usually have subject and predicate parts inside and they can be subdivided into following two types.

- Subject + Verb: 胃/下垂 (gastroptosis), 地/震 (earthquake)
- Verb + Subject: 搬运/工 (porter), 裁判/员 (referee)

- **Verb-object**

Words of this kind usually have internal verb and object parts. They also contain two types as shown below.

- Object + Verb: 党/代表 (representative of party), 手/套 (gloves)
- Verb + Object: 理/发 (haircut), 反/政府 (anti-government)

- **Verb-complement**

Words of this kind usually have a verb part and a complement part, which shows the result, direction or aspect of the verb’s action. These words have the following two types.

- Verb + Verb: 跑/出来 (running out of), 打发/掉 (get rid of)
- Verb + Adjective: 染/红 (dyeing red)

- **Parallel-combination**

Words of this kind have a coordinate structure where the meanings of constituents are equivalent, similar, related or opposite. And these coordinate constituents always show the properties of the word’s original meaning. In Chinese, most of these words are two-character words, though there exist some cases of long words. The examples are shown as follows.

- 开/关 (switch), 学/习 (learning)
- 中/日/韩 (China, Japan and Korea)
- 农/林/牧/副/渔
(farming, forestry, animal husbandry, side-line production and fishery)

- **Head-modification**

Words of this kind are those that have a head part and a modification part, which shows the property of the head part. This kind of words is most common in Chinese synthetic words. Usually the right part is the head and the left part is the modifier.

- 放大/器 (amplifier), 冲印/店 (print shop), 汽车/站 (bus stop)

The above five kinds internal syntactic relations are the most basic ones among the component of Chinese synthetic words. As shown in the above examples, most Chinese synthetic words consist of two-character or three-character words that only have one internal syntactic relation inside. However, when the word length gets longer, these relationships will be embedded with each other to produce a quite complicated form.

5. Exceptions

Besides compound words and morphologically derived words, there still exist some types of words in real Chinese text, which need more discussion on whether they belong to synthetic words or not. However, we can use some other methods to deal with these kinds of words, such as time expression extraction, named entity recognition, etc. These kinds of words are shown in the following.

- **Abbreviations**

Abbreviations are expressions that have a short appearance while standing for a longer term. For example:

- 中共 → 中国共产党 (Communist Party of China)

- 国资委 → 国有资产监督管理委员会
(State-owned Assets Supervision and Administration Commission)

- **Factoids**

Factoids are expressions that indicate date, time, number, money, score or range. These kinds of expressions have a large variation in their appearance. For example:

- date: 2008.1.30, 2008 年 1 月 30 日, 二零零八年一月三十日
- time: 5:30, 5 点 30 分, 五点三十分, 五点半 (five thirty)
- number: 2345, 二三四五, 贰叁肆伍
- money: 三元五角六分, 三块五毛六 (3.56 yuan)
- score: 1:3, 1 比 3, 一比三 (one to three)
- range: 1 到 5, 一到五 (from one to five)

- **Idioms, proverbs, allegorical sayings and fragments of poems**

These kinds of expressions usually consist of more than three characters and always have a special meaning in Chinese. Sometimes they look like sentences while actually having compact meanings like words. For example:

- idioms (成语): 门可罗雀 (sparsely visited)
- proverbs (俗语):
巧妇难为无米之炊 (One cannot make bricks without straw)
- allegorical sayings (歇后语):
哑巴吃黄连 (有苦说不出)(unable to express bitter feelings)
- poem fragments (诗词片段):
先天下之忧而忧 (be the first to bear hardships)

6. Research targets

As indicated from Section 2.3 and Section 2.4, for a typical Chinese synthetic word, a fully internal structure analysis result should include both construction

structure information and syntactic relationship information. In Chapter 4, we conduct machine learning experiments to predict both kinds of categories for three-character synthetic word. But in Chapter 5 and Chapter 6, we only conduct experiments to predict construction structure of synthetic words, because of the following two reasons.

- when native Chinese speakers analyze a synthetic word, they intend to first divide the word into parts before determining the internal syntactic relation of it. Thus in our work, we feel natural and reasonable to conduct the same process as human being do, which is structure first and relation second.
- we use machine learning methods in our experiment for analysis. Though we have some Chinese language resources at hand, none of them contain any syntactic information that we can learn from. Without this knowledge of syntactic information as features, we can hardly predict the internal relationship of synthetic words at present.

Chapter 3

Related Work on Synthetic Word Analysis

In Chinese language processing community, although there are a lot of researches on the morphological analysis for written text, few are specialized in analyzing the internal structure of words, especially Chinese synthetic words. However, in actual morphological analysis systems, every research group has their own way of dealing with synthetic words by using their segmentation standard. Until now, there have been the following researches that did similar work on the analysis of Chinese synthetic words.

1. A Chinese synthetic word corpus

Huang et al. [15] use multi-layered segmentation standards in corpus annotation. In other words, they described an annotation process of a corpus, which considered some kind of information of synthetic words, rather than conducting a detailed synthetic word analysis research. During the creation process of their corpus, they apply three tags to control the segmentation level of synthetic words. These three tags are w0, w1 and w2, which stand for ‘faithful’, ‘truthful’ and ‘graceful’ respectively. For example, for a synthetic word ‘北京市安全厅 (security agency of Beijing city)’ , this tagging method analyzes the word as shown in Figure 3.1.

This kind of tagging method can describe synthetic words’ structure within

Figure 3.1. Synthetic word tagging approach

Word 北京市安全厅 (security agency of Beijing city)
Tags <w2>
<w1><w0> 北京 </w0><w0> 市 </w0></w1>
<w1><w0> 安全 </w0><w0> 厅 </w0></w1>
</w2>

small (two) levels. But if the target synthetic word has much more complicated structure, using these tags will not be enough to hold all necessary structure information.

Another thing is, because they create a hand-annotating corpus, it is easy for them to define internal syntactic relationship categories and include this information when doing the annotation work. But unfortunately, they failed to do that.

2. Rule-based approach on synthetic words

The second related research was done by Microsoft Research. In [34], they create a customizable segmentation system for Chinese morphologically derived words. Basically their goal is the same as ours: to let morphological analysis system generate more flexible output with different kinds of granularities. Before developing the system, the morphologically derived words are first divided into the following five kinds:

- Reduplication
讨论讨论 (discuss), 看看 (take a look)
- Affixation
副 主席 (vice chairman), 警察局(police station)
- Directional and resultative compounding
走进去 (walk in), 看清楚 (see clearly)

- Merging and splitting
国内外 (domestic and foreign), 洗了澡 (took a bath)
- Named entities and factoids
诺罗敦 · 西哈努克 (Norodom Sihanouk)
世界贸易组织 (World Trade Organization)
四百五十 (four hundred and fifty-six)

Then from the internal features of these kinds of morphologically derived words, they define about fifty formation rules that are used to testify the generated segmentation parts. Finally, based on these pre-defined rules, the customizable morphological analysis system uses a parameter driven method, which can divide synthetic words into different levels of word components according to the needs of different NLP applications. The parameters here are simply binary values that indicate the system to treat a word as a single word or a compound word.

Although this system achieves higher score than other systems that do not have synthetic analysis, it mainly uses a rule-based method for this task.

3. Chinese multiword chunking

The third related research was done by Tsinghua National Laboratory for Information Science and Technology of China. In [41], they defined a task named ‘Multiword chunking’ to automatically analyze the external function and internal structure of the multiword chunks (MWC) in a sentence. In order to do this task, they first divide multiword chunk into five types shown in Table 3.1.

Table 3.1. Types of multiword chunk in Chinese

np	noun chunk	mp	quantity chunk
vp	verb chunk	sp	space chunk
ap	adjective chunk	tp	time chunk

Then they define the following five kinds of relation tags inside these multiword chunks.

ZX: modifier-head relationship

PO: verb-object relationship

SB: verb-compliment relationship

LH: coordinate relationship

LN: chain hooking relationship

Finally, they propose an efficient rule acquisition algorithm to automatically extract formation rules of multiword chunks from their corpus (Tsinghua Chinese Treebank), which has the annotated information of chunk types and internal relation tags. A piece of this corpus is shown below.

[tp-ZX 长期/t(long time) 以来/f(since)], /w 他/r(he) 为/p(for) 维护/v(safeguard) [np-ZX 世界/n (world) 和平/n(peace)] 的/u [np-ZX 崇高/a(lofty) 事业/n(undertaking)] [vp-PO 倾注/v(devote) 心血/n (painstaking)] , /w 作出/v(make) 了/u 卓越/a(outstanding) 的/u 贡献/v(contribution)。

POS tags used in the sentence: t-time noun, f-direction, r-pronoun, p-preposition, v-verb, n-noun, u-auxiliary, a-adjective, d-adverb, w-punctuation.

As you can see from this corpus, the biggest difference between their work and ours is that their purpose is to extract formation rules of phrases, while our purpose is to analyze the internal information of synthetic words. Because phrases and synthetic words are not on the same level of segmentation in Chinese text, we cannot adopt their method in our own research. And the high expense of using this corpus also avoids us to use the information of phrase relation in this corpus, which might be useful to indicate the internal relationship of synthetic words.

4. Analysis on Japanese medical terminology

The fourth related research is about analysis on Japanese Medical Terminology. In [35], Yamada and Matsumoto use character-wise dependency relation parsing to analyze the internal structures of Japanese Technical Terms. Though their experiment target is not Chinese language, Japanese Technical Terms have the same characteristics like Chinese: long un-separated characters are usually used as one single unit. The parsing result is shown in Figure 3.2.

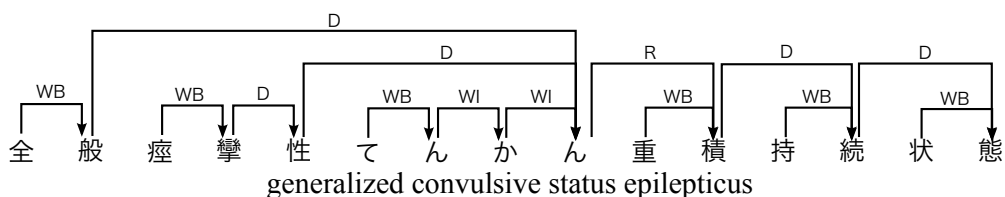


Figure 3.2. Dependency parsing on Japanese Medical Terminology.

They actually achieve a quite high accuracy 88.2% for whole structure of synthetic words. But in their experiment, a lot of gold standard feature has been used, including MeSH (Medical Subject Heading). Actually, MeSH is a kind of very detailed category information specifically developed for Medical text recourse management.

Though they did not conduct many experiments to see what is the most useful feature in their experiment, we believe that MeSH plays a very important role in their dependency parsing process.

5. Structural analysis on English noun phrase

The last related work is an analysis on English noun phrases. In [30], Pitler et al. use Google’s web-scale N-gram corpus as resources for computing features for structure analysis. But this time their target language is English. As one can imagine, since English does use space to delimit word boundaries, their task is not as complicated as ours.

In their work, they use point wise mutual information as features to determine the probability of whether an internal words list can be used as an internal part.

And then use that information to parse the whole English phrase' s structure.

As explained above, because English is quite different from Chinese in the characteristic of word, we cannot adopt their approach directly. But we think the features computed from large-scale resources might be a good clue for parsing Chinese synthetic words too, we use the same methods on Chinese in Chapter 6, but with specific customized feature set and parsing algorithm.

6. Other related researches

There are also some other similar works done both in English and Chinese language domain. In [5], Jenny Rose Finkel et al claimed that English nested named entity recognition had been entirely ignored due to technological rather than ideological reasons. And they used a discriminative constituency parser (CRF-CFG parser) to recognizing those nested named entities in English. Though they did a successful work on this task, we can barely use their method because their research target is quite different from ours in following two points: first, English has explicit word boundary while Chinese not; second, they emphasized on recognizing nested named entities while we want to parse internal structure for common words. Furthermore, the CRF-CFG parser used in their work is a lexicalized parser for English sentence, so we can not use it directly in our research. However, the approach they used in building the CRF-CFG parser is quite impressive and we may try to use the same approach when building our parser in future as an extension of our research.

Another related work is [37]. In this paper, Hai Zhao et al. proposed a character-level dependency scheme to represent primary linguistic relationships within a Chinese sentence. In other words, they transformed word segmentation task for sentence into a Chinese character-level dependency-parsing task. Though their task is quite different from ours, the basic idea is the same: trying to divide Chinese character sequence into some kind of structure without considering word boundary. We actually adopt some parts of their methods in our own research, which will be further described in Chapter 5.

Chapter 4

Parsing Three-character words with Adjacency Model

In order to specify the consistency of Chinese segmentation standard used in our morphological analysis system and fertilizing the information of our dictionary, we apply machine learning methods to analyze internal information of synthetic words, based on the categories of compound words and morphologically derived words introduced in Chapter 2.

Generally, synthetic word analysis means to get the knowledge of internal parts of word with all length. However, since Chinese has only one type of character (Hanzi) and long words can be broken into small parts with specific construction pattern, we think it is reasonable to first focus our attention on the shortest unit of synthetic words. Because short synthetic words has the same construction pattern as longer ones, if we find a way to analyze short synthetic words, we can use them as the base information to long synthetic words with the same approach.

1. Two basic parsing models for three-character synthetic words

Speaking of short synthetic words, two-character words and three-character words are good start point to analyze. However, though two-character words in Chinese

could have internal syntactic relations between the two characters, most of them are often used as single words by Chinese native speakers intuitively. And most of them have already been registered as lexical entries in the morphological analysis system at each institution. Furthermore, breaking every two-character words into single character is not practically useful because that is only way to separate two-character synthetic words.

For these reasons, we start our research on three-character words, and then enlarge the span to long synthetic words.

The main approach to analyze the structure of three-character words has been to compute association statistics between pairs of characters and then choose the bracketing (left or right branching) that corresponds to the more highly associated pair. Generally there are two models:

- adjacency model
- dependency model

Under the adjacency model, the branching decision is made by comparing the associations between two adjacent pairs. For example, let A,B,C stand for independent characters and ABC stands for a valid word. What the adjacency model does is to compare the association strength between AB and BC, and finally select a stronger pair for branching.

In contrast, the dependency model compares the association strength between AB and AC, and finally selects AB for left branching or AC for right branching.

In this Chapter, we conduct experiment on three-character synthetic words based on adjacency model. And In Chapter 5 we try to use dependency model to analyze them.

2. Statistical models and tools

2.1 Supervised and unsupervised machine learning method

Supervised and unsupervised learning are two common machine-learning methods frequently used in natural language processing. The difference between them is that the actual status for each piece of training data is available in supervised

learning, while the categories of training data are unknown in unsupervised learning. Usually, supervised learning can be seen as a classification task based on the knowledge of training data and unsupervised learning can be seen as a clustering task without any prior knowledge.

In this chapter, since we define synthetic words based on both internal syntactic relation and internal morphological structure, we mainly use supervised machine learning method to classify synthetic words into their corresponding categories based on the knowledge learnt from the training data.

2.2 Mutual information

In information theory, mutual information is often defined as the holding between random variables. But in natural language processing, mutual information could be seen as a measurement of binding relations between two words in a large text. And it is often used as a standard for discovering collocations. Thus the originally defined mutual information between particular words w_1 and w_2 is shown in as the following.

$$I(w_1, w_2) = \log_2 \frac{P(w_2 w_1)}{P(w_2)P(w_1)} \quad (4.1)$$

In the following section, we will use mutual information frequently as a measurement of binding condition of internal component pairs of word. While the common use of mutual information is to find two parts whose connection is tight, we will use it in an opposite way to find out parts whose connection is loose.

2.3 Support vector machines

Support vector machines, which are widely used in natural language processing community, are binary classifiers that search for hyperplanes with the largest possible margin between positive and negative samples. Suppose we have a set of training data for a binary class problem: $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)$, where $\mathbf{x}_i \in R^n$ is a feature vector of the i th sample in the training data and $y_i \in \{+1, -1\}$ is the label of the sample. Its goal is to find a decision function which accurately predicts y for an unseen \mathbf{x} . An SVM classifier gives a decision function $f(\mathbf{x})$ for an input vector \mathbf{x} where

$$f(\mathbf{x}) = \text{sign}(\{ \sum_{\mathbf{z}_i \in SV} \alpha_i y_i K(\mathbf{x}, \mathbf{z}_i) \} + b) \quad (4.2)$$

$f(\mathbf{x}) = +1$ means that \mathbf{x} is a positive member, and $f(\mathbf{x}) = -1$ means that \mathbf{x} is a negative member. The vectors \mathbf{z}_i are called support vectors, which receive a non-zero weight α_i . Support vectors and the parameters are determined by solving a quadratic programming problem. $K(\mathbf{x}, \mathbf{z})$ is a kernel function which maps vectors into a higher dimensional space. We use a polynomial kernel of degree 2 given by $K(\mathbf{x}, \mathbf{z}) = (1 + \mathbf{x} \cdot \mathbf{z})^2$.

Though SVM is originally designed for binary pattern classification, we can also use it in multi-class pattern recognition problems. In this case, there are two ways to do the classification: one-against-all and one-against-one. Both of them use a combination of binary SVM and a decision strategy to decide the class of the input pattern.

We use LIBSVM [14] in the remaining part of this dissertation. LIBSVM is an integrated software for support vector classification, (C-SVC, nu-SVC), regression (epsilon-SVR, nu-SVR) and distribution estimation (one-class SVM). Since it supports multi-class classification using a one-against-one pattern, we use it for classifying the synthetic words into the categories of compound words.

3. Dataset

In order to conduct experiment on Chinese synthetic words, corpus or other language resources are necessary. Actually, the only language resources we have are Chinese GigaWord and Chinese dictionary of ChaSen system.

Chinese GigaWord First Edition

Chinese GigaWord First Edition, produced by Linguistic Data Consortium (LDC), is a comprehensive archive of newswire text data that has been acquired from Chinese news sources by the LDC over several years. Inside, it contains two distinct internal sources of Chinese newswire, which are Central News Agency of Taiwan and Xinhua News Agency of Beijing. The total data of GigaWord reaches

to 286 files, approximately 1.5GB in compressed SGML form using a very simple, minimal markup structure.

We use Chinese GigaWord for computing mutual information of word internal parts in the following section. A sample of Chinese GigaWord is shown in Figure 4.1.

```
<DOC id="XIN19901231.0007" type="story">
<HEADLINE> 湖北 1990 年利用外资工作取得进展</HEADLINE>
<DATELINE> 新华社武汉 1 月 1 日电</DATELINE>
<TEXT>
<P>(记者罗辉、实习生刘凉)1990 年，湖北省利用外资工作取得进展，全年利
用外资的合同金额达 3.14 亿美元。</P>
<P>1990 年，全省还批准“三资”企业 94 家，比 1989 年增加 24 家。至此，
全省已批准“三资”企业 277 家，总投资为 5.53 亿美元。</P>
</TEXT>
</DOC>
```

Figure 4.1. Sample of Chinese GigaWord

As you can see from Figure 4.1, there are a lot of tag information in Chinese GigaWord. Because we only want to compute mutual information between internal parts of words, the tag information in Chinese GigaWord is useless. Thus, we remove all the tag information from it and prepare a bunch of clean texts for computing mutual information later.

Chinese Dictionary of ChaSen

The Chinese version of ChaSen is a two-layer morphological analysis system based on Hidden Markov model. As the base of this segmentation system, it has two dictionaries that contain common words and parts-of-speech information, which are necessary in morphological analysis.

The first dictionary, which is quite small, has a total number of 33,474 entities. The words inside are mainly extracted from Chinese Treebank, which is frequently used in Chinese natural language processing. And the second dictionary, which

has a quite large scale number of entries, is an extension of the first one by adding entries using out-of-vocabulary words detection and referring on other language resources described in [12]. At present, the large dictionary, which is called ChaSen dictionary in the following, has a total number of 135,767 entries.

Because it is difficult for present morphological analysis systems to resolve the internal structures of Chinese synthetic words having more than two characters, we examine the distribution of these words in the above two dictionaries. The results are shown in Table 4.1, in which “A” means the original number of entries in each dictionary and “B” means the number of entries after removing NR (proper nouns such as name, location, organization, etc.).

Table 4.1. Distribution of words more than two characters

		3 characters	4 characters	≥ 5 characters
small dic (33,474 entries)	A	5,237	1,713	331
	B	4,241	1,424	69
ChaSen dic (135,767 entries)	A	25,379	6,424	5,732
	B	16,380	3,437	985

As you can see from this table, three-character words have the largest number, which are also the most common short synthetic words we should focus on.

Since both dictionary of ChaSen and Chinese GigaWord do not have internal information for word, in order to conduct our experiment, we have to annotate our own. By using the system which will be introduced in Chapter 7, we first annotate 1,000 three-character words with their internal construction structure and syntactic relationship information. Table 4.2 and 4.3 show the annotation results.

Table 4.2 gives us the information that most three-character synthetic words (83.0%) have an internal structure of left branching, which means that they may have suffix, from the morphological structure aspect.

In Table 4.3, although about 5.4% of the 1,000 three-character words are single-morpheme words, we still see that words belonging to Head-modification category occupy the largest part (84.2%) in synthetic words from the internal syntactic relation aspect.

Table 4.2. Distribution of MDW words' categories in 1,000 words

Flat branching (infix)	0.5%
Left branching (suffix)	83.0%
Right braching (prefix)	9.0%
Merging	1.5%
Reduplication	0.2%

Table 4.3. Distribution of compound words' categories in 1,000 words

Subject-predicate	4.8%
Verb-object	2.0%
Verb-complement	3.2%
Parallel-combination	0.2%
Head-modification	84.2%
Single-morpheme word	5.4%

Since most three-character Chinese words (about 92.0%) have the internal structure of left branching (suffix) or right branching (prefix) formation, it is obvious that we should analyze Head-modification words with frequently used suffixes in three-character words at the beginning of our research. And we can get a list of possible affixation characters from this process.

Furthermore, because reduplication words tend to have fixed structures which make them easy to recognize, we make a survey of reduplication words in our current Chinese dictionary of ChaSen and the results are shown in Table 4.4.

Table 4.4. Number of words in reduplication structure

AA	AXA	ABB	AAB	AABB	ABAB	AXAY	XAYA
547	37	102	68	74	4	151	36

4. Experiment

We conduct three experiments in this section. They are all targeted on the 1,000 annotated three-character synthetic words. The former two experiments try to classify these words into categories based on their morphological structure, which are left, right or flat branching. The last experiment tries to classify them into categories based on their internal syntactic relationship.

4.1 Experiment one: using mutual information only

Based on the adjacency model described in Section 4.1, we assume that if a three-character word ABC has an internal structure of left branching AB/C, then the relationship between A and BC will be much tighter than the relationship between AB and C. On the other hand, if a three-character word ABC has an internal structure of right branching A/BC, then the relationship between AB and C will be much tighter than the relationship between A and BC.

For example, for word ‘冲印店 (print shop)’ which has left branching structure, the relationship between ‘冲’ and ‘印店’ is tighter than the relationship between ‘冲印’ and ‘店’. This is because ‘印店’ alone is not a valid word which cannot be divided from 冲印店 independently, while ‘冲印’ is a valid word which can be used independently.

With this simple idea, we try to use mutual information of each pair for comparison, whose result is used to decide the categories of internal morphological structure for a word. Since we don’t have any other useful resources except for Chinese GigaWord (CGW), we have to use CGW to compute mutual information between internal parts of words.

In the actual experiment, if we have a word ABC, in which A, C, AB, BC are all independent word entries in our dictionary, we compute the mutual information MI_{RB} for A and BC, and the mutual information MI_{LB} for AB and C, where the RB and LB stand for right branching and left branching. Then we apply the following two rules to predict the final branching type.

- if $MI_{RB} < MI_{LB}$, word ABC has right branching (prefix) structure

Because the relation between A and BC is more independent than the relation between AB and C, which means it is more possible that A and some

other two-character word XY could form a valid word AXY . So the possibility of word ABC having a right branching structure is greater than the possibility of it having a left branching structure.

► **if $MI_{RB} > MI_{LB}$, word ABC has left branching (suffix) structure**

Because the relation between AB and C is more independent than the relation between A and BC , which means it is more possible that C and some other two-character word XY could form a valid word XYC . So the possibility of word ABC having a left branching structure is greater than the possibility of it having a right branching structure.

The result of classification is not good. At last, we have 676 out of 920 words (73.5%), which are classified correctly by only looking at the internal mutual information in Chinese GigaWord.

After analyzing the result, we find that most incorrect ones are words having left branching internal structure, but wrongly classified to have right branching internal structure. As one can imagine, for word ABC , when both left branching (AB) and right branching (BC) are valid words, and they both have quite high mutual information computed from Chinese GigaWord, it is difficult and not proper to determine the internal structure only by using mutual information. We need more features to assist us in classification.

4.2 Experiment two: using SVM classifier

In this experiment, besides mutual information, we include more features for help. And this time, we use SVM classifiers to learn those features from training set, and make decision on testing set. The features we use show in Table 4.5.

The first two features, characters and part-of-speech of each internal part, are extracted from the Chinese dictionary of ChaSen. If an internal part is not registered in this dictionary, then we simply set its part-of-speech to null, which will be ignored in the training and testing process of the SVM classifier. The last two features, frequency and mutual information are computed by using Chinese GigaWord.

After dividing training set and testing set, we run the SVM classifier several times by changing the mod of frequency we actually use in computing. And the

Table 4.5. Features for classification on morphologically derived words

for three-character word ABC	
characters	A, C, BC, AB, ABC
POS	pos(A), pos(C), pos(BC), pos(AB), pos(ABC)
frequency in GigaWord	fre(A), fre(C), fre(BC), fre(AB), fre(ABC)
mutual information	$MI_{RB}(A-BC)$, $MI_{LB}(AB-C)$

overall accuracy, precision and recall are shown in Table 4.6, from which we can see the frequency mod 2000 gets the best result here.

Table 4.6. Results of SVM on classifying left and right branching internal structure

Frequency mod	Accuracy	Branching	Recall	Precision
10,000	92.93%	Right	33.33%	85.71%
		Left	99.40%	93.22%
5,000	93.48%	Right	38.89%	87.50%
		Left	99.40%	93.75%
2,000	94.02%	Right	38.89%	100.0%
		Left	100.0%	93.79%
1,000	92.39%	Right	22.22%	100.0%
		Left	100.0%	92.22%

Because the results in Table 4.6 dose not consider the existence of two-character words in system dictionary, we then add the following two features by referencing the original dictionary and run the SVM classifier again.

- in the case of three-character string ABC
- features:
 - if ABC has right branching internal structure, see whether BC is a valid word

- if ABC has left branching internal structure, see whether AB is a valid word

And the final results are shown in Table 4.7. In Table 4.7, the first row (old) shows the result of frequency mod 2000 in Table 4.6, and the second row (new) shows the result after adding the above features. Both of them use the frequency mod 2000 in the actual computing.

Table 4.7. Results after adding more features by referencing CGW

Experiment	Accuracy	F value	Right Branching		Left branching	
			Recall	Precision	Recall	Precision
Old	94.02%	0.56	38.89%	100.0%	100.0%	93.79%
New	94.57%	0.67	55.56%	83.33%	98.80%	95.35%

As we could imagine, adding more useful features does improve the overall accuracy in classifying morphologically derived words.

However, this result is quite unbalanced because there are only a few instances having right branching internal structure both in training set (9.78%) and testing set (9.78%). This is the reason for why the recall is low in classifying instances with right branching (prefix).

Though there are some words that were wrongly categorized, we still get an overall accuracy of 94.57%, which would be much higher if we recursively use SVMs for classification. We believe that this method could classify morphologically derived words quite efficiently if we add some more rules for recognizing merging and reduplication words. However, this experiment does not take merging structure into consideration and it mainly use golden-standard feature like character, POS. In other words, this approach may not be applicable to words more than three characters. We will further discuss this problem in Chapter 6.

4.3 Experiment three: classification on compound words

As introduced in Chapter 2, we know that for most Chinese synthetic words, we can annotate them with two kinds of tags according to their internal syntactic

relation and internal morphological structure respectively.

After conducting previous two experiments, we get a dataset of three-character synthetic words with the knowledge of their internal structure: left or right branching. Therefore, though we have few useful syntactic resources at hand, we try to classify these words into the categories from the aspect of compound words (internal syntactic relationship) too. Different from the last experiment, compound words have five categories rather than left and right branching. Thus this time we used a multi-class version of SVM classifier to divide those words into the compound word categories based on their internal syntactic relation. The features we use this time are shown in Table 4.8.

Table 4.8. Features for classification on compound words

for three-character word ABC with left branching (or right branching)	
POS	pos(ABC), pos(AB), pos(C) (or pos(ABC), pos(A), pos(BC))
structure information	left branching (or right branching)

The first feature, part-of-speech, is a re-annotated result of target words based on the rules shown in Table 4.9. The re-annotation is conducted because the original POS tag use the POS set of Penn Chinese Treebank, which has small categories specific for chunking. For example, verb can have the following four tags: VA, VC, VE, VV. (The meanings of these tags are listed in Appendix A.) However, what we need here is a coarse-grained tag set, only used to determine whether a word is a noun or a verb etc.

Table 4.9. POS re-annotation rules for compound words

Subject-predicate	NN+VV, VV+NN
Verb-object	VV+NN, NN+VV
Verb-complement	VV+VV, VV+JJ, AD+VV, JJ+JJ, AD+JJ, JJ+P
Head-modification	NN+NN, JJ+NN, NR_LOC+NN, VV+NN, NN+VV

By using the re-annotated version of POS, we get the following result from

the multi-class SVM classifier.

Table 4.10. Results after part-of-speech modification

	Precision	Recall	F
Verb-object	100.0%	80.00%	0.89
Verb-complement	85.71%	85.71%	0.86
Noun-modification	100.0%	100.0%	1.00

As shown in Table 4.10, the result of each category is quite unbalanced. For example, the SVM classifier classified all Head-modification words correctly, but the results of other categories are not that good. We believe that it is because the dataset itself is quite unbalanced, since we have a large number of words with Head-modification category, while there are only a small number of examples in other categories. However, from this experiment, we know that it is not good enough to determine the internal relation categories of words by only using the current dataset and feature list. We'd better focus our attention on analyzing the internal structure of synthetic words, rather than try to specify their internal syntactic relationship among different internal parts.

5. Summary

In this Chapter, we conduct three experiments for three-character synthetic words mainly based on adjacency model, which makes branching decision by comparing the associations between two adjacent pairs. We achieved a quite good overall accuracy on deciding the branching type of three-character synthetic words by using mutual information comparison and other golden standard features like POS etc.

However, those experiments do not take merging structure into consideration because the merging pattern is really rare in the 1,000 annotated target words. Because a lot of golden standard features are used in experiments, this approach may not be applicable to words more than three characters. We will try to avoid using golden standard feature and apply a rather general approach for parsing

synthetic words in Chapter 6.

In the end of this Chapter, we try to predict internal syntactic relationship by using multi-class classifiers and gain a quite unbalanced result. This encourages us to first focus our attention on analyzing the internal structure of synthetic words, rather than try to specify their internal syntactic relationship among different internal parts, because we do not have any Chinese syntactic language resources available.

Chapter 5

Parsing Three-character Words with Dependency Model

1. Analysis using dependency parsing approach

In the previous Chapter, we mainly conduct experiments on three-synthetic words based on adjacency model. In this Chapter, we use dependency model as the underlying model to analyzing synthetic words.

In [19], they use dependency information between English words to analyze structure of English noun compounds. However, comparing to English, Chinese is written based on characters rather than words, thus we conduct our experiments based on character dependency information.

We believe there are four advantages by using character dependency parsing approach.

- Chinese is born to be character sequence rather than word sequence, so it feels more natural if we conduct dependency parsing based on characters not words. And according to [35], they gain better result on character dependency parsing than word dependency parsing for doing the task of Japanese medical terminology analysis. We believe it would also be the case when parsing Chinese.
- Character dependency inside words are similar as words dependency inside phrase or sentence. Though we use character dependency parsing method,

we could take advantage of word information in the upper level of parsing.

- Since we only recognize characters at the bottom level when doing character dependency parsing, we have no need to pay attention to ordinary word boundaries, which might be a possible error source if we use word dependency parsing.
- Character dependencies can transform to a tree structure easily. Figure 5.1 shows the transformation process.

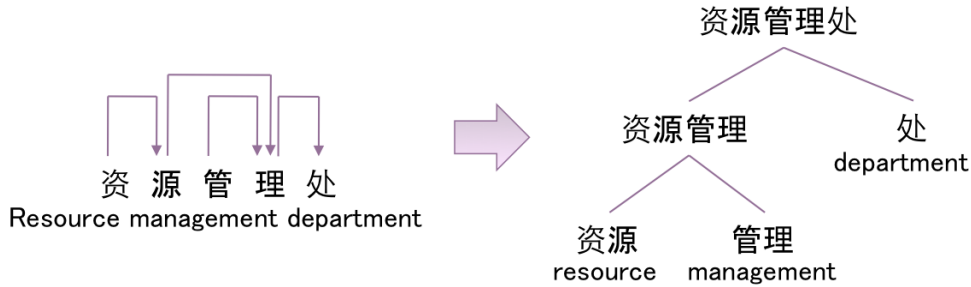


Figure 5.1. Transformation from dependency to tree structure

2. MST Parser

Dependency parsing is the process of creating dependency trees for input sentences by looking at its internal dependency grammar. At present, there are two commonly used parsers available: MaltParser [27] and MSTParser [23]. In this Chapter, we choose MSTParser as our tool to conduct character dependency parsing. MSTParser is a non-projective dependency parser that treats the parsing problem as the search for the highest scoring maximum spanning tree in the complete directed graph over the input sentence. By using MSTParser, a dependency tree is computed as the sum of the scores of adjacent edge pairs. In this process, weight learning is performed using an extension of the Margin Infused Relaxed Algorithm (MIRA) [3], an online learning algorithm particularly well-suited to structured classification problems.

Ordinarily, MSTParser expects a sentence and a list of features as input, and generate dependency tree based on those information. Since in Chinese,

character dependency inside words are similar as words dependency inside phrase or sentence. So we assume MSTParser will treat synthetic words as the same way as sentences, except for the length of input string becoming short.

3. Experiment

3.1 Transformation from dependency tree to structure tree

Though we use character dependency parsing to analyze target synthetic words, its output, ‘dependency tree of synthetic word’, is not what we want. What we want in the end is the internal structure tree of each synthetic word. Therefore, we have to define a way to transform generated dependency edges to internal structure tree. Because the final internal structure tree could be very complicated due to various kinds of construction process, we need to define a set of edge labels to satisfy all the structure patterns that synthetic words have.

After carefully examining all kinds of structure patterns of synthetic words, we use the label set shown in Table 5.1 to represent the transformation factors.

Table 5.1. Label set for transforming from dependency tree to structure tree

for every edge between two characters	
Label	Representing structure
B	branching structure between parts
C	coordinate structure between parts
WB	internal word’s beginning part
WI	internal word’s other parts
MR	reverse modifier in left merging structure

By using these edge labels as representation, we can transform a dependency tree to a structure tree accordingly, such as Figure 5.2.

In Figure 5.2, ‘中 -C-> 日 -C-> 韩’ represents a coordinate structure for word ‘中日韩’. ‘围 -WB-> 棋 -B-> 赛’ means word ‘围棋赛’ have a branching structure with internal parts of ‘围棋’ and ‘赛’, and ‘围棋’ is a two-character synthetic word acting as the left branching part.

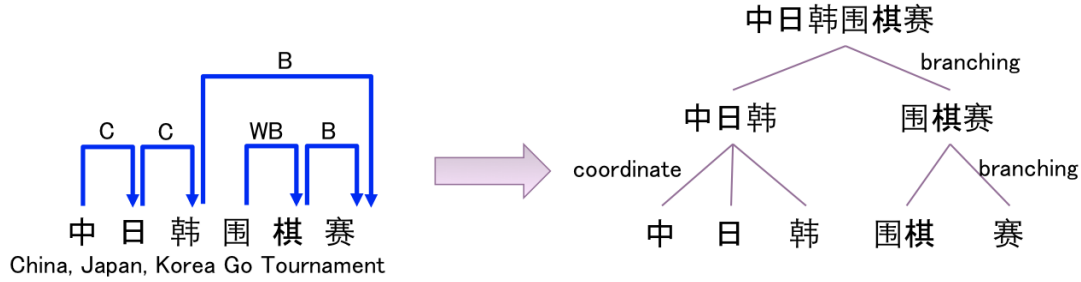


Figure 5.2. Transformation from dependency to tree structure using edge labels

In fact, the label set shown in Table 5.1 is carefully developed, so it can be used to transform almost any kind of dependency tree to structure tree for synthetic word. However, since our target is three-character synthetic words, we had better understand what kinds of dependency trees they possibly have. Figure 5.3 shows all possible dependency trees for three-character synthetic words, with merging structure taken into consideration this time.

3.2 Dataset and feature

In the following experiment, we use the 1,000 hand-annotated three-character words, described in Chapter 4, as dataset to conduct the dependency parsing method. In order to use MSTParser, we arrange the three-character word dataset in the format like Figure 5.4.

As shown Figure 5.4, we use four kinds of features to predicate the corresponding dependency tree: the character itself, POS, label and position. Here POS means all possible part-of-speech of the corresponding character. Label means the tag we use to specify edge type in the dependency tree, which we just defined. For the root character, in this case is ‘者’, because there is no edge coming from it, we simply tag it as ‘Root’. The last feature we use is position, which is the position of semantic head of each character. Again, because there is no head for root word, we tag its position as zero.

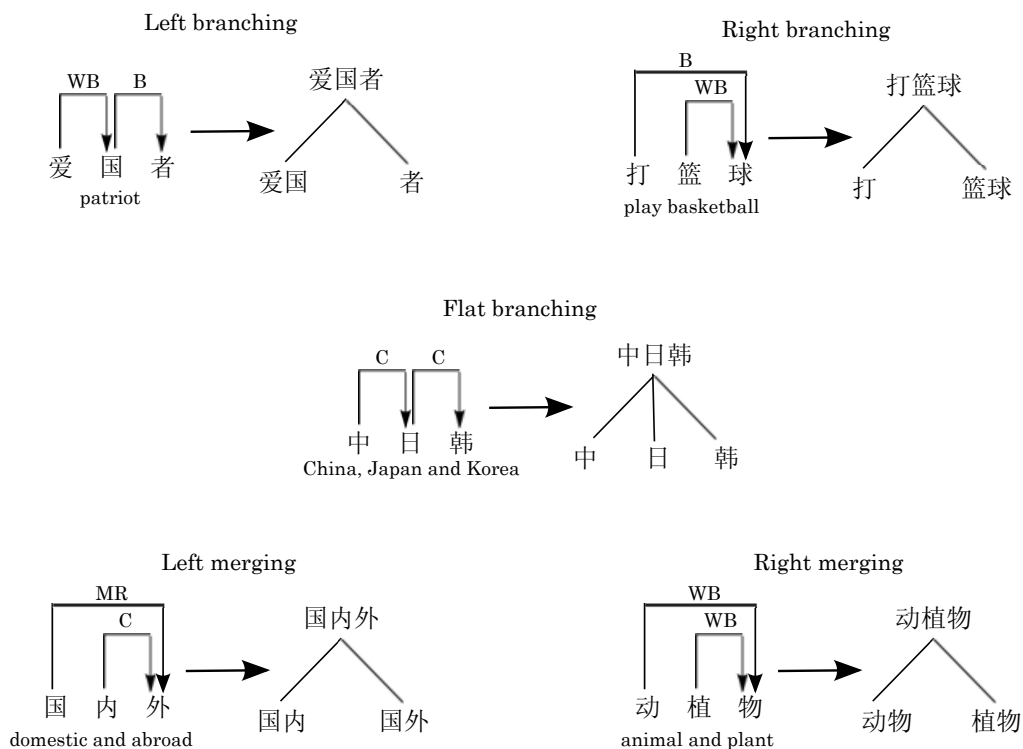


Figure 5.3. Three-character dependency trees

Character:	爱	国	者
POS:	NN NR-PER-GIV VV	NN NR-ORG NR-PER-FAM NR-PER-GIV	NN
Label:	WB	B	ROOT
Position:	2	3	0

Figure 5.4. Three-character word dataset for MSTParser

3.3 Experiment result and discussion

With those features described above, we conduct the dependency parsing method on 1,000 annotated three-character words. Although we can represent all possible structures as dependency trees by using our proposed label set shown in Figure 5.3, we only experiment on words with left branching and right branching structure. The reason is the cases that words have merging or flat branching structure are so rare in our 1,000 annotated words that we do not expect the parser can

distinguish them from other structures. The results we get from MSTParser are shown in Table 5.2.

Table 5.2. Experiment results of MSTParser on three-character synthetic word

	Accuracy	Completely Correct	Tree Baseline
Exp.1	0.917	0.878	0.911
Exp.2	0.922	0.881	0.901
Exp.3	0.918	0.871	0.894

In Table 5.2, Exp.1-3 means three different experiments with different training and testing set split on the target 1,000 three-character words. The first column ‘Accuracy’ means the overall label accuracy for every character node in all synthetic words. The second column ‘Completely Correct’ means the accuracy of the whole structure tree after transforming from generated dependency trees. The last column shows the baseline of complete structure tree in each experiment’s testing set.

As indicated in Table 5.2, though the overall label accuracy of each character is not bad (91.5% in average), the complete structure tree accuracies are lower than the baselines in all three experiments, which indicates it may not be an effective way to parsing synthetic words by using dependency analysis. We find the following may be the reasons that cause this unsatisfactory result.

- The number of Chinese synthetic word structures is quite unbalanced in test set, with left-branching structures more than 90%. This is the same reason why we can not get better performance in Chapter 4 too. The three-character synthetic words are just so typical that we can not learn much more structure from them expect the left-branching type, especially in a small dataset of 1,000 words.
- All the POS information is extracted from our dictionary of Chinese ChaSen morphological system. It is a golden standard feature and most of part-of-speech in this dictionary were either generated by ChaSen automatically or annotated by several human annotators. Because there was no cross

validation work was done on these POS, they may not be very accurate for further usage in other systems.

- Since we conduct our dependency parsing experiment on character unit, we do not use any kind of information for words in MSTParser. And we believe the lack of syntactic information resources also pulls down the performance of dependency parser.
- MSTParser is a dependency analyzer for general purpose. It takes all the edge combinations among each node into consideration in the background parsing process.

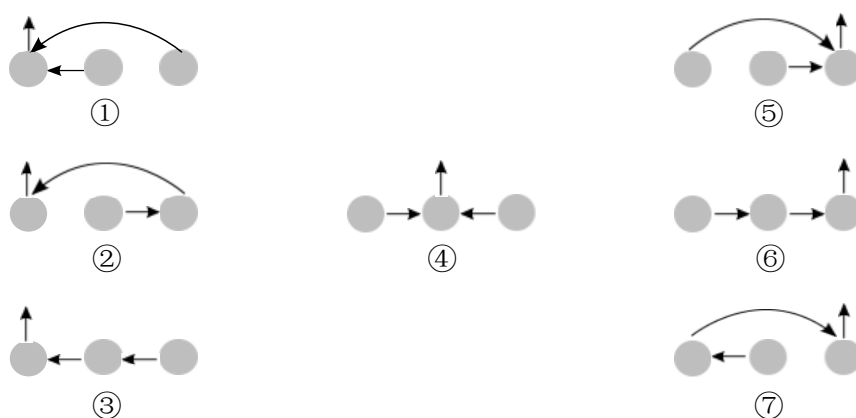


Figure 5.5. All possible combinations considered by MSTParser

As indicated in Figure 5.5, for three-character synthetic words, if we use MSTParser to generate character dependency tree, it will consider all nine possibilities in the background parsing process. However, since we only consider left or right branching structure in this experiment, only type ⑤ and type ⑥ are fit in this case. So what MSTParser do in the background is trying to distinguish two types out of nine, not just selecting between those two types as we expected. This maybe a very important reason why the complete structure accuracy is lower than baseline.

4. Summary

In this chapter, we try to analyze internal structure of three-character synthetic words based on dependency model. We develop a label set for the dependency model, which can be used to transform a system generated dependency tree to a structure tree that we really expect. By using a general dependency parser, MSTParser, we conduct experiment on 1,000 annotated three-character synthetic words, with left branching and right branching structure taken into consideration. The experiment result is lower than baseline because of the special characteristic of three-character Chinese synthetic words and the general mechanism of MST-Parser.

Chapter 6

Tree-based Parsing Approach for All Synthetic Words

1. Tree-based parsing approach

Synthetic word construction is the most common phenomenon in Chinese language. Native Chinese speakers tend to make new words from common characters by using certain construction patterns based on syntactic relationship among these characters. Despite the importance of synthetic words, most existing Chinese sentence parsers do not parse synthetic words. It is because in the main training corpus of these parsers, synthetic words are represented either as single units or in flat structures.

In Chapter 4 and Chapter 5, we try to analyze three-character synthetic words with both adjacency model and dependency model. From the analysis results, we find that three-character synthetic words have some specific characteristics comparing to ordinary synthetic words, and those approaches can not be easily applied on long synthetic words with more than three characters.

In this Chapter, we will focus on parsing long synthetic words, which have more than three characters. When we analyze three-character synthetic words, their structures are quite simple and we can transform the parsing task into a classification task, like we do in Chapter 4. But in the case of long synthetic words, we cannot simplify the task into a classification problem because the structures get complicated while the word lengths become long.

As described in Chapter 1, our goal is to take a long synthetic word as input and produce a parse tree as output. Since two-character words are commonly used as single words in Chinese, the lowest level of this parse tree should only contain two-character synthetic words and single-morpheme words, which are introduced in Chapter 2. An example of parse tree is shown in Figure 6.1.

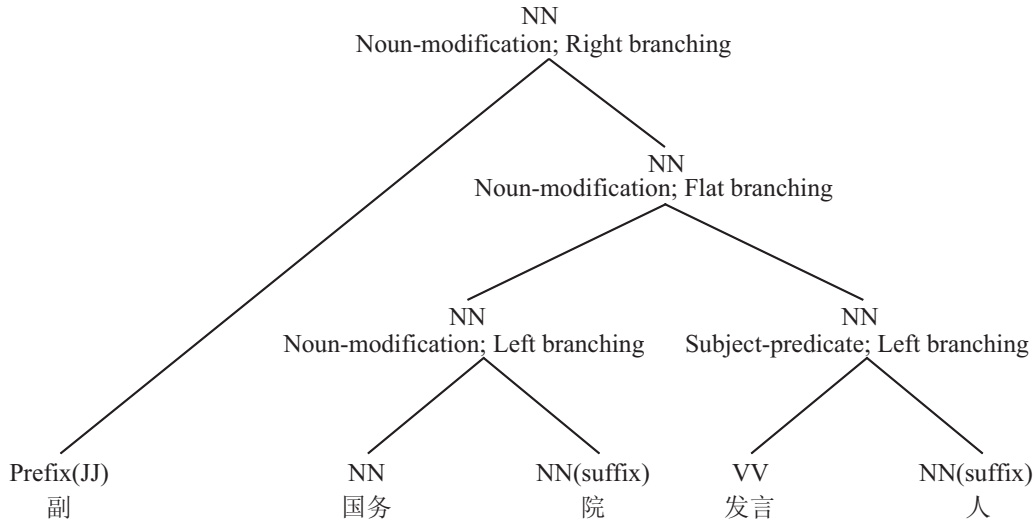


Figure 6.1. An example of parse tree of long synthetic word

In practice, because we do not have any useful Chinese syntactic language resources at hand, it is quite difficult for us to predict the internal syntactic relationship between internal units, as described in Chapter 4. Normally, predicting the internal relationship among internal parts on each level of synthetic word need to be done with the prior knowledge of what the internal structure is. Therefore, at present we first concentrate on predicting the internal construction structure of synthetic words.

However, parsing synthetic word for its construction structure is also a challenging task because the same part of speech sequence can be parsed differently depending on the specific words involved. For example, the two synthetic words shown in Figure 6.2 both have the same POS sequence: NN / NN / NN, but their structure is completely different.

So in this case, we need common lexical statistics in order to parse synthetic words like these. And those lexical statistics should be extracted over a large set

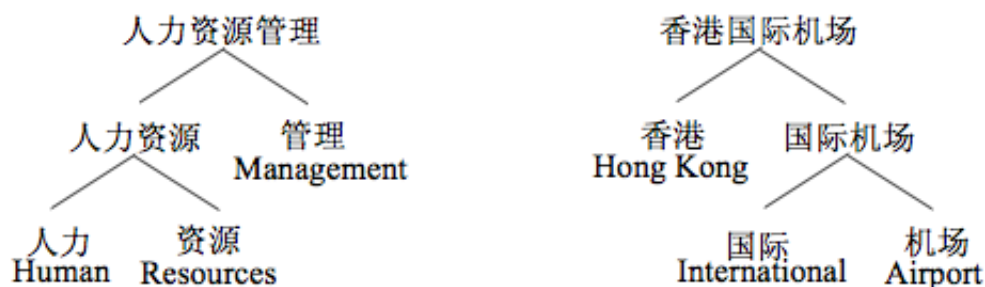


Figure 6.2. Different parsing tree with same POS sequence

of text to avoid sparsity.

Another issue is, being different from English, Chinese synthetic word parsing is more difficult because we do not have the prior knowledge of words. Since all we have is character sequence, we have to construct parsing tree while predicting word boundaries. Of course one can use a dictionary for referring whether a characters sequence is an internal part, but that kind of gold-standard information will affect parsing result, depending on how the dictionary being used is constructed. And sometimes, using word information from dictionaries will not help at all. For example in word shown in Table 6.2, all of its regular internal character combinations are concrete words.

Table 6.1. Internal part candidates for a special word

word	天文学界 (astronomy academic society)
internal part candidates	天文 (astronomy), 文学 (literature), 学界 (academic society) 天文学 (astronomy), 文学界 (literati)

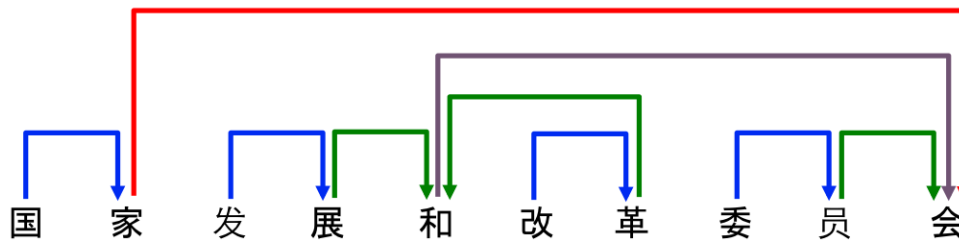
From these problems, we can imagine parsing long synthetic words to a tree structure is a difficult task.

2. Top-down approach vs. bottom-up approach

Generally speaking, there are two way of analysis approach considering the final tree structure output. One is top-down approach; the other one is bottom-up approach. Figure 6.3 shows an example of them.



a. top-down approach



b. bottom-up approach

Figure 6.3. Top-down approach and bottom-up approach for analyzing long synthetic words.

In Figure 6.3, we can see that top-down approach separates long words into different parts in every level from top (word itself) to bottom (smallest internal units). On the contrary, bottom-up approach takes its way from the lowest level (each character in the target words), and pops up to upper level when it finds possible combinations among internal units.

The question is what approach we should take when doing analysis on long Chinese synthetic words.

If we use the top-down approach, apparently we should segment long words into shorter ones, just like the segmentation for sentence, with the length of the target string getting much shorter.

However, though the syntactic relationships among units inside a long word are similar to those relationships among words inside a sentence, the whole syntactic structures of long word and sentence are quite different, which will make parsing words in a top-down manner quite difficult.

Furthermore, as shown in the bold rectangle in the left side of Figure 6.3 a, those character consequences are not actual words. They are just some kinds of intermediate production in the process of top-down parsing. Since they cannot act as actual words, trying to specify ‘word boundary’ of them is pointless.

On the contrary, if we take bottom-up analysis method, things will become much more natural. Apparently, in bottom-up parsing process, we do not need to be aware of what are words or where are word boundaries; we just scan the target character in the sequence from left to right, and find proper combination candidate units for upper level. We believe there are three good reason for doing analysis in this way.

- We can take many candidates into consideration. Theoretically, we could take all the internal part candidates into consideration. For a word with length N , the possible candidates’ number with possible character length is $N - (L - 1)$, where L is the character length of that candidate. For example, word ‘激光打印机 (laser printer)’ has five possible candidates with one character (激, 光, 打, 印, 机) and four possible candidates with two characters (激光, 光打, 打印, 印机), etc.
- We can finally take merge structure into consideration. Since we start from scanning characters, we can use custom scanning rule to create special internal part candidate for merging structure. For example, when parsing word ‘中小学教师 (middle and primary school teacher)’, we can create special internal candidates (中学, 小教, 学师) to provide information to determine whether there is a merging structure or not. We will further explain this in next section.
- Since word level information is not necessary to be prior knowledge in

bottom-up parsing, these information or features may in turn aid the parser to get more confidence on whether internal part candidates could be words and not.

On the basis of these reasons, we take the bottom-up approach when analyzing long synthetic Chinese word in this Chapter. Our idea is simple: take a sequence of characters as input and generate a parse tree as output. In detail, our approach has two steps:

- The first step is using SVM learning machines to verify every possible internal character sequence, and generate a margin score for each candidate indicating how confident this sequence can be used as an internal part in structure.
- The second step is to use a customized CYK parser to compute all possible tree structures bottom-up with score values generated using SVM for every possible internal candidate. And finally the tree structure with highest score will be considered as the correct parsing tree.

3. Parsing algorithm

3.1 Original CYK parsing algorithm

In the original form of the CYK (Cocke-Younger-Kasami) algorithm, what it does is to determine whether a string can be generated by a given context-free grammar and, if so, how it can be generated. This algorithm employs bottom-up parsing and dynamic programming. The standard version of CYK operates on context-free grammars given in Chomsky normal form.

In the theory of computation, the importance of the CYK algorithm is the fact that it constructively proves that it is decidable whether a given string belongs to the formal language described by a given context-free grammar, and the fact that it does so quite efficiently.

In our case, since we do not want to use word level POS information as prior knowledge, we do not have a context-free grammar for synthetic words based on some kind of syntactic grammar. Instead, we simply consider all synthetic words

as character strings, and use CYK algorithm to constructively prove whether a certain internal structure is an acceptable internal tree structure based on possible internal part candidates' information. Figure 6.4 shows an example of the construction process.

激光打印机 (laser printer)

0	1	2	3	4	
激	激光	激光打	激光打印	激光打印机	0
	光	光打	光打印	光打印机	1
		打	打印	打印机	2
			印	印机	3
				机	4

Figure 6.4. CYK construction table for a synthetic word.

In Figure 6.4, every cell in the CYK table has a sub-sequence of the original character string. And for each cell, there are one or several construction candidate patterns related to it. For the word shown in Figure 6.4, the construction candidates for the original CYK algorithm are list in Table 6.2.

In the actual parsing process, by constructing such table like Figure 6.4, we can get every possible tree candidate and select the best one from them based on the weight information in each cells of CYK table.

So the reason why we use plain CYK algorithm is quite clear:

- It is a bottom-up parsing approach which we describe and select in previous section.
- Comparing to parsing synthetic words using dependency parsing method, using CYK can directly generate the final parsing tree, there is no need to do the transformation work.
- With a little customization, this algorithm can basically consider any possible character sequence in the construction process of structure tree.

Table 6.2. Internal part candidates for a special word

Characters	Index	construction candidate patterns
激	[0, 0]	[0, 0]
光	[1, 1]	[1, 1]
打	[2, 2]	[2, 2]
印	[3, 3]	[3, 3]
机	[4, 4]	[4, 4]
激光	[0, 1]	[0, 0] + [1, 1]
光打	[1, 2]	[1, 1] + [2, 2]
打印	[2, 3]	[2, 2] + [3, 3]
印机	[3, 4]	[3, 3] + [4, 4]
激光打	[0, 2]	([0, 0] + [1, 2]), ([0, 1] + [2, 2])
光打印	[1, 3]	([1, 1] + [2, 3]), ([1, 2] + [3, 3])
打印机	[2, 4]	([2, 2] + [3, 4]), ([2, 3] + [4, 4])
激光打印	[0, 3]	([0, 0] + [1, 3]), ([0, 1] + [2, 3]), ([0, 2] + [3, 3])
光打印机	[1, 4]	([1, 1] + [2, 4]), ([1, 2] + [3, 4]), ([1, 3] + [4, 4])
激光打印机	[0, 5]	([0, 0] + [1, 4]), ([0, 1] + [2, 4]), ([0, 2] + [3, 4]), ([0, 3] + [4, 4])

Nevertheless, CYK algorithm has shortcomings too. The biggest problem is the computational cost. For a string of length n , the computational efficiency is $O(n^3)$ in the worst case.

However, the good thing is that our targets are words, and not sentences. Normally, n will not be too large, so the performance will not be too bad in most cases. And we could use some approaches like k-best methods to improve the efficiency in some cases. We further talk about this in the next subsection.

3.2 Customized CYK parsing algorithm

As described in Chapter 2, Chinese synthetic words has a special phenomenon called ‘merging’, where two words create a truncated (or merged) form using common internal part when they are concatenated. Though some merging-structure

synthetic words can be seen as branching structures with a internal part having parallel or coordinate meaning, in this research we consider them all as words with merging structure due to the reasons described in Chapter 2. Comparing to left branching or right branching, there are not so many words having merging structure in Chinese, therefore it is hard to recognize them using classification method mentioned in Chapter 4.

Although the number of merging structure is not many in our dataset, we want to find a way to represent it in the parsing process of long synthetic words. But the original CYK construction process does not consider merging structure. For example, for character sequence ‘激光打’, CYK cell $[0, 2]$ in Table 6.2, the complete internal structure candidates we expect are shown in Table 6.3.

Table 6.3. Complete internal structure candidates for character sequence ‘激光打’

Type	Characters	Indexes
right branching	激 + 光打	$[0, 0] + [1, 2]$
left branching	激光 + 打	$[0, 1] + [2, 2]$
left merging	激光 + 激打	$[0, 1] + ([0, 0] + [2, 2])$
middle merging	激光 + 光打	$[0, 1] + [1, 2]$
right merging	激打 + 光打	$([0, 0] + [2, 2]) + [1, 2]$

By comparing Table 6.2 and Table 6.3, we can see that original CYK parsing algorithm only considers left branching and right branching, leaving the merging structures untouched. The reason for this lies in the construction characteristic of CYK parsing algorithm, and we can change this behavior by customizing it. The prototype program of our customized CYK algorithm is shown in Figure 6.5.

As indicated in Figure 6.5, by including those ‘special_cases’ in CYK algorithm, we can almost add any kind of special structure that original CYK can not generate to CYK cells for later complete tree structure computation. And since these special structure are only considered at the corresponding level, the whole computational cost of CYK algorithm will not change.

Theoretically, merging structures also exist in words with more than three

```

for length = 2 ..... n
  for row = 0 ..... (n - length)
    for column = row + (length - 1)
      for k = row ..... (column - 1)
        ## ordinary CYK pair
        cell[row][column].structs << cell[row][k], cell[k + 1][column]
      end
      ## special cases like merging
      cell[row][column].structs << special_cases
    end
  end
end
end

```

Figure 6.5. Prototype of customized CYK algorithm

characters. However, since those cases are really rare and the most common merging structure are normally seen in three-character span, we only implement three-character merging structures in this customized CYK algorithm, which are left merging, middle merging and right merging.

Finally, because we conduct experiments on long synthetic words this time, and there are some cases where the synthetic words' lengths are quite long. In order to verify the efficiency of our customized CYK algorithm, we conduct a little test based on synthetic word length. When doing a complete computation for all candidate trees and finally select a best one from them, the CYK algorithm's benchmark is shown in Table 6.4.

Table 6.4. Candidate tree numbers generated by CYK when doing complete search

Word Length	Candidate Num.	Word Length	Candidate Num.
10	21,368	14	527,984
11	84,134	15	22,721,790
12	335,839	16	94,062,075
13	1,356,010	17	use more than 128GB memory

As shown in Table 6.4, the computing resource requirement of CYK algorithm will become unacceptably large while the target synthetic word length getting longer. To make our system faster, we employ a 10-best restriction in CYK algorithm, which means we only consider the 10 best structure candidates at every construction level according to the weights of candidate trees.

4. Structure annotation of synthetic words

As described in Chapter 4, we only have two language resources at hand for conducting experiments: Chinese GigaWord and dictionary of ChaSen. And there is no gold standard resource with internal word structure information specified for doing synthetic word analysis. Since we want to apply supervised machine learning methods in following experiments, this kind of data is essential in the whole process.

This kind of situation drives us to annotate our own gold standard dataset for experiments. In Chapter 4, we have already annotated 1,000 three-character synthetic words. But obviously that is not enough for general-purpose experiments. Furthermore, because we want to analysis long synthetic words which have more than three characters this time. We need gold standard resources for long synthetic words too.

Therefore, we further annotated 10,000 three-character synthetic words and 1,000 long synthetic words with more than three characters. In order to make sure those long synthetic words are general words rather than idioms, poems, etc. we randomly extract them from the terms of Chinese version of Wikipedia.

Though there are a lot of details and standards we defined in the process of annotation, normally the annotation process is like the following:

1. Determine whether a target word is a synthetic word or not. If it is a single-morpheme word, go on to the next target word; if it is a synthetic word, follow step 2.
2. Split the target word into internal parts from top level to bottom level, and tag what kind of split category each level belongs to: merging or branching.

3. After finishing annotating the internal structure, try to specify which syntactic relationship each split belongs to, according to definition of compound word category introduced in Chapter 2.

In the actual annotation process, we make use of the lexical management system we created to annotate the internal structure and save them in database with a proper format. This system will be further explained in Chapter 7. At last, we have 11,176 synthetic words annotated by three annotators and the distribution of these words is shown in Table 6.5.

Table 6.5. Distribution of annotated synthetic words

Character number	3	4	5	6	7	≥ 8
Word count	10049	205	294	216	201	201

5. Experiment

5.1 Detailed parsing method

The idea of our approach is quite simple. We take long un-separated character sequence as input, and produce a complete parsing tree as output. To accomplish this task, we create a system that takes the following two steps in the whole process.

- **Step one:**

When we are given an input character sequence, our system uses a supervised learner to predict the score of any possible particular contiguous subsequences given the entire input string as context. This means that rather than inclusively inserting brackets to hold possible connected characters, we compute whether adjacent characters can be separated at every possible position and level.

For example, the synthetic word ‘副国防部长 (vice secretary of defense)’ could be translated into several different classification problems shown in

Table 6.6. Possible internal candidates for 副国防部长

Current bracket position	Target internal part
[副国] 防部长	副国
[副国] 防部长 ★	副防
副 [国防] 部长	国防
副 [国防] 部长 ★	国部
副国 [防部] 长	防部
副国 [防部] 长 ★	防长
副国防 [部长]	部长
[副国防] 部长	副国防
副 [国防部] 长	国防部
副国 [防部长]	防部长
[副国防部] 长	副国防部
副 [国防部长]	国防部长

Table 6.6, with each determining the inner character sequence is whether a possible internal part or not.

Note there are several ★ marked in Table 6.6. Since Chinese synthetic word could be merging structure within three character span, but merging structure can not be described only using brackets here, we use ★ to indicate them. In practice, the corresponding background classifier will determine whether this character sequence, which is the very common merging internal part, is valid or not. For example, in the above table, character sequence ‘副国防’ could be merging structures like [副防, 国防] or [副国, 副防], the common merging part ‘副防’ will be taken into consideration in this case.

The output margin scores of these classifiers can be easily put into a full structure parser as weights for each character sequence. In following experiment, we use LIBSVM, a Library for Support Vector Machines, to do this learning task, and then convert the SVM output to scores to indicate how likely each possible character sequence could be a valid internal part.

- **Step two:**

After we gain the margin scores for each sub character sequence, indicating whether it could be a valid internal part or not, we put them as features in the customized CYK parser to conduct complete search for the best tree with highest score.

The customized CYK parser is the one we introduced in last section. Besides the original construction patterns, we include special cases representing merging structure in three-character level. And we add a 10-best construction tree threshold on each level to avoid the CYK parser taking up too much computing resources.

5.2 Features preparation

Compute N-gram data

In following experiments, rather than using gold standard feature like POS as features for SVM classifier, we want to use more general lexical statistics features to let SVM compute more confident probabilities in Step one for each possible character sequence.

As described in Chapter 4, the largest Chinese language resources we have at hand is Chinese GigaWord, so we decide to compute n-gram data from it. The version of Chinese GigaWord we used is about 1.5GB in size and has 12 years newspaper texts from both Taiwan and Mainland China. We assume any basic internal parts of Chinese synthetic word do not have more than 5 characters. And we use the suffix array method described in [20] to quickly get n-gram count with length 1 to 5.

Suffix array is a data structure designed for efficient searching of a large text. The data structure is simply an array containing all the pointers to the text suffixes sorted in lexicographical (alphabetical) order. Each suffix is a string starting at a certain position in the text and ending at the end of the text. Searching a text can be performed by a binary search using the suffix array. For example, for a sentence like ‘abracadabra’, the method to compute its n-gram is shown in Table 6.7.

In the upper part of Table 6.7, the first column is the suffix list of original

Table 6.7. Example of suffix array for character sequence ‘abracadabra’

suffix array		head character cutting number				
original	sorted	1	2	3	4	5
abracadabra	a	a				
bracadabra	abra	a	ab	abr	abra	
racadabra	abracadabra	a	ab	abr	abra	abrac
acadabra	acadabra	a	ac	aca	acad	acada
cadabra	adabra	a	ad	ada	adab	adabr
adabra	bra	b	br	bra		
dabra	bracadabra	b	br	bra	brac	braca
abra	cadabra	c	ca	cad	cada	cadab
bra	dabra	d	da	dab	dabr	dabra
ra	ra	r	ra			
a	racadabra	r	ra	rac	raca	racad
n-gram result			ab:2	abr:2	abra:2	abrac:1
			ac:1	aca:1	acad:1	acada:1
		a:5	ad:1	ada:1	adab:1	adabr:1
		b:2	br:2	bra:2	brac:1	braca:1
		c:1	ca:1	cad:1	cada:1	cadab:1
		d:1	da:1	dab:1	dabr:1	dabra:1
		r:2	ra:2	rac:1	raca:1	racad:2

character sequence, and the second column is the sorted version. If we want to compute n-gram, we just cut the heading character by number n , then re-count the frequency of every cut heading. This will give us the n-gram count result we want shown in the lower part of Table 6.7.

By using this method, we can quick compute 1 to 5 n-gram data from Chinese GigaWord. Even it only has length 1 to 5 n-gram, the computed size of data becomes to 17GB, and we use these count as base to compute classification features.

Feature used in experiments

According to [30], the more position-specific our features were, the more effectively we could parse synthetic words. Here the position means the relative distance from the rightmost character. Therefore, for each character sequence shown in Table 6.6, we use a carefully selected matrix of features related to the current position of bracket. By using features in this way, the feature set for every sub character sequence is actually disjoint, this is equivalent to use multiple SVM learning machines, with each one determine each internal part combination on specific bracket position.

For every sub character sequence, we include four kinds of features.

- **Current internal part’s length and current bracketing position**

These two features are quite intuitive. Since most of long synthetic words have left-branching structure. We use the relative distance from rightmost of character sequence as bracketing position.

- **Point-wise mutual information (PMI)**

The reason to use PMI is simple. For instance, consider a bracket from character 5 to character 6, if character 6 and character 7 have a higher PMI value than character 5 and character 6 does, then the proposed bracket is unlikely, otherwise it can be considered as correct bracket. All PMI values are computed using the following function with probabilities generated from Chinese GigaWord n-grams.

$$PMI(x, y) = \log \frac{p(xy)}{p(x)p(y)}$$

As explained before, the parsing performance will be better if we link every PMI feature to current considering bracket position. So here we carefully arrange these PMI features in a matrix related by current bracket position and use that matrix as features for SVM leaning machines. In the actual experiment, because we use a span of two to five character length, using this matrix means we are applying a joint model of both adjacent approach and dependency approach, which introduced in Chapter 4.

In [30], for word of length n , they include all combination of 2 internal word parts' PMI. But we select different set of PMI values here because we do not have prior knowledge of Chinese word boundary inside target character sequence. Since we assume the base internal part of Chinese synthetic word is no longer than five characters, we include PMI feature shown in the following Table 6.8.

Table 6.8. PMI feature selection for SVMs

Current considering character string		
Window size	Features	
$C_{h-5}C_{h-4}C_{h-3}C_{h-2}C_{h-1}(C_hC_{h+1}C_{h+2}C_{h+3}C_{h+4}\dots C_{t-5}C_{t-4}C_{t-3}C_{t-2}C_{t-1})C_tC_{t+1}C_{t+2}C_{t+3}C_{t+4}$		
2	left outside	$C_{h-3}C_{h-1}$ $C_{h-2}C_{h-1}$
	left overlap	$C_{h-2}C_h$ $C_{h-1}C_h$ $C_{h-1}C_{h+1}$
	left inside	C_hC_{h+1} C_hC_{h+2}
	right inside	$C_{t-3}C_{t-1}$ $C_{t-2}C_{t-1}$
	right overlap	$C_{t-2}C_t$ $C_{t-1}C_t$
	right outside	C_tC_{t+1} C_tC_{t+2}
	left outside	$C_{h-3}C_{h-2}C_{h-1}$ $C_{h-3}C_{h-2}\dots h-1$ $C_{h-3}\dots h-2C_{h-1}$ $C_{h-3}\dots h-2C_{h-3\&h-1}$ $C_{h-2}\dots h-1$ $C_{h-3}\dots h-2C_{h-2}\dots h-1$
3	left overlap	$C_{h-2}C_{h-1}C_h$ $C_{h-2}\dots h-1C_h$ $C_{h-1}C_hC_{h+1}$ $C_{h-1}C_{h\dots h+1}$
	left inside	$C_hC_{h+1}C_{h+2}$ $C_hC_{h+1}\dots h+2$ $C_{h\dots h+1}C_{h+2}$ $C_{h\dots h+1}C_{h\&h+2}$ $C_{h+1}\dots h+2$ $C_{h\dots h+1}C_{h+1}\dots h+2$
	right inside	$C_{t-3}C_{t-2}C_{t-1}$ $C_{t-3}C_{t-2}\dots t-1$ $C_{t-3}\dots t-2C_{t-1}$ $C_{t-3}\dots t-2C_{t-3\&t-1}$ $C_{t-2}\dots t-1$ $C_{t-3}\dots t-2C_{t-2}\dots t-1$
	right overlap	$C_{t-2}C_{t-1}C_t$ $C_{t-2}\dots t-1C_t$ $C_{t-1}C_tC_{t+1}$ $C_{t-1}C_{t\dots t+1}$
	right outside	$C_tC_{t+1}C_{t+2}$ $C_{t\dots t+1}C_{t+2}$ $C_{t\dots t+1}C_{t\&t+2}$ $C_{t+1}\dots t+2$ $C_{t\dots t+1}C_{t+1}\dots t+2$

As shown in Table 6.8, we not only consider each combination between two characters, but also taking word boundary into consideration and including every possible word formation around left or right bracket. Actually, besides two and three character window size, we also include four and five character window size in experiments.

For all these PMI features, if it is computable, we use it, otherwise we include one of the following two binary features:

$$p(xy) = 0 \text{ or } p(x)p(y) = 0$$

- **Lexical information**

We have annotated the internal structure for 10,000 three-character words. Since most three-character synthetic words have left branching or right branching structure, we can generate a list of possible prefix and suffix with their frequencies from this annotated resource. We include these frequencies as features because of the following reason: if these affixations appear at the two side of left or right bracket, then that bracket may have a high possibility to be a good one, and vice versa.

Because we want to build a model for all common synthetic words using machine learning method, we do not use any kind of gold-standard features like characters, POS, etc.

5.3 Experiments on three-character synthetic words

As usual, we first conduct experiments on three-character synthetic words because they are the most (10,027 out of 11,144) in the annotated dataset. The distribution of our annotated three-character synthetic words is shown in Table 6.9.

As shown in Table 6.9, the left branching structure is very common in three-character words' internal structures. And as a result, it gives a high baseline to compare.

In this case, because the target words are three-character words, there is no need to include features more than two character window size. Table 6.10 gives the classification and parsing accuracy.

Table 6.9. Distribution of annotated three-character synthetic words

Dataset (10,027)	Training set (8,027)	Testing set (2,000)
Left branching	7055 (87.89%)	1760 (88.00%)
Left merging	15 (0.19%)	2 (0.10%)
Middle merging	6 (0.07%)	1 (0.05%)
Right merging	88 (1.10%)	18 (0.90%)
Right branching	810 (10.09%)	210 (10.50%)
Flat branching	53 (0.66%)	9 (0.45%)

Table 6.10. Classification and parsing results of three-character synthetic words

Baseline		Result	
Classification Acc.	Parsing Acc.	Classification Acc.	Parsing Acc.
91.9%	88.0%	93.2%	90.0%

Here ‘classification accuracy’, which is the output of SVMs, shows the proportion of how many sub-character sequences are predicted as valid internal parts of target synthetic word. And ‘parsing accuracy’, which is the output of CYK parser, shows by using the score values converted from SVMs’ results, how many complete structure of synthetic word is correctly predicted.

In Table 6.10, ‘baseline’ means all three-character synthetic words have left-branching internal structure. Since left branching is the largest part for three-character synthetic words, our system has little chances to learn from other kinds of structure in training process. This kind of result is in expectation.

5.4 Experiments on all synthetic words

In experiments on all annotated synthetic words, we first use features in windows size two, and add features in other window size accordingly. The results are shown in Table 6.11.

As shown from the above table, when parsing words with more than three-

Table 6.11. Classification and parsing accuracy for all synthetic words

Word length:	≥ 4				≥ 3	
Features:	PMI		PMI+Lexical		PMI	
PMI window size	Accuracy					
	Clas.	Pars.	Clas.	Pars.	Clas.	Pars.
PMI_2	91.6%	32.0%	90.0%	35.0%	92.5%	83.7%
PMI_{2+3}	92.0%	40.0%	90.4%	39.0%	92.5%	84.0%
PMI_{2+3+4}	91.8%	39.5%	89.7%	37.5%	92.3%	84.2%
$PMI_{2+3+4+5}$	91.5%	39.5%	89.3%	39.0%	92.0%	84.1%

characters, using features of window size three give the parsing accuracy a boost from 32.0% to 40.0%, but features of window size four and five are not that useful because it causes drop in both classification and parsing result. And adding lexical information which indicating affixation is not helpful in this case.

Furthermore, one can notice that when parsing words with more than three characters, the parsing accuracy is quite low comparing to the corresponding classification accuracy. So we further investigate the performance for each internal part length when parsing accuracy is 40.0% using feature set PMI_{2+3} only. Table 6.12 shows the investigation result.

Table 6.12. Classification performance with different internal parts' length

Internal parts' length	Accuracy	Precision	Recall	F1
2	93.2%	89.2%	82.2%	0.855
3	90.0%	78.6%	47.8%	0.594
4	86.1%	78.0%	32.5%	0.458
5	93.4%	55.6%	45.5%	0.500
≥ 6	95.2%	65.5%	38.8%	0.487

As shown Table 6.12, while the target internal parts' lengths get longer, the recall of classification falls quickly. We believe this is main reason that causes the parsing result for whole structure getting very low. Because in our customized

CYK parsing algorithm, despite its lower structure's score is fairly high, if one formation candidate's score gets very low, it will quickly fall out of consideration in the upper construction process, as we take a 10-best restriction in CYK parsing algorithm.

After detailed examination of the test result, we find the following issues in experiment process may be the cause of low parsing result, and we will try to improve them in the future.

- The Chinese GigaWord that we use to compute n-grams count and generate our PMI features is quite large, but may not be large enough in this case. Because what we want here is common lexicon statistics, the low recall of longer internal part candidate indicates that we should use web-scale text data like the work of [30].
- There are some synthetic words that are really hard to tell what kind of structure it has. So the annotation work of our dataset may be inconsistent in some cases because different annotators' opinion may differ on very controversial structures. Even in the test result, we can still find some parsing results that look correct, but annotated in another way.
- We find that synthetic words containing transliteration internal part are always tagged with wrong structure on that part. This is normal because we did not use any kind of feature to indicate a part is a foreign word, so the CYK parser tends to construct them from the bottom using characters one by one.
- Words with parallel structure are always tagged as wrong in testing result. This is because we did not include any kind of word boundary information in the parsing process, and our current CYK parser can not handle structure like 'A + B + C', because single characters' scores are set to 1 (which is the highest) and this kind of structure will always get the highest score, which in turn will eliminate all other structure candidates.
- We also find that the PMI features we used are mainly focused on the position of bracket, which indicates whether a part should be outside of that bracket. But we ignored combination inside the bracket, which might

provide strong clue to indicate whether a part should be inside of that bracket.

6. Summary

In this Chapter, we concentrate on conducting bottom-up tree-based parsing method for all synthetic words, especially for words with more than three characters. We use a rather large dataset, GigaWord, to compute Chinese n-gram data and then compute point-wise mutual information between distinct internal parts of words from this n-gram dataset. We try to use common statistical information like PMI as features for parsing, rather than gold standard information like POS. Finally we use two-step parsing method on all synthetic words and get complete parsing accuracy from it. Though the final result of complete tree parsing accuracy is not very good due to various reasons, we believe the parsing approach itself is right, because rather complicate structures of words like ‘副新闻发言人 (vice spokesman)’ are parsed correctly. At last, we list some reasons that may cause the low accuracy of parsing and propose possible solutions to them.

Chapter 7

An Extendable Lexicon Management System

1. Background

Along with the rapid development of natural language processing in recent years, the scale of language resources used in research becomes larger and larger. This is especially the case for lexical dictionaries, which are essential for many systems, because all the time lexical information is expanding not only vertically but also horizontally. Figure 7.1 shows this phenomenon in the case of Japanese dictionary.

ID	単語	品詞	活用型	活用形	
1525	あほらしい	形容詞 - 自立	形容詞 - イ段	基本形
1526	あまい	形容詞 - 自立	形容詞 - アウオ段	基本形	
1527	あまえる	動詞 - 自立	一般	基本形	
1530	あまがける	動詞 - 自立	五段 - ラ行	基本形	
216363	補する	動詞 - 自立	サ変 - 一スル	基本形	
216913	宝物殿	名詞			
.....					

Figure 7.1. An example of lexicon information expansion

Here, 'Vertically' means that there will always be new entries added into sys-

tem, such as domain-specific terminologies and ordinary out-of-vocabulary words, while ‘horizontally’ means that there will always be new information added to all or some part of lexicons to support succeeding researches.

Normally, ordinary lexicon management systems can handle vertical expansion without problem, but few of them can handle horizontal expansion properly. It is because horizontal expansion includes a lot of sparsity in the dataset and requires dynamic structure modification in the underlying database.

Furthermore, lexical annotation is known as an extremely time and labor consuming task in all language domains. Since we want to annotate internal structures of synthetic words of Chinese, Japanese and possibly other languages in our own research, it is difficult to do this task on existing systems in that they do not have a proper design to hold that annotation results for tree structure. In order to fit all the above needs, we developed an extendable lexicon management system named *Cradle*¹ to ease both the annotation task and the management of large-scale lexicons.

2. Aims and tools

Aims

With the aim of making the *Cradle* system more powerful and user-friendly, we have several principles laid out for the system before development, which are:

- It should be based on the Web for easy access using ordinary web browsers.
- It should have a user-management functionality, which can be used to set which user can see what kind of information, such as dictionary types etc.
- It should be extendable in managing lexicon, both vertically (to increase word entries) and horizontally (to increase properties for the word)
- It should have a proper way to handle the annotation task for lexicon, especially for the task of synthetic word annotation

¹running at <http://dahlia.naist.jp/cradle>

- It should be extendable in source code to manage other possible language domains besides Japanese and Chinese
- It should have a relatively quick response when searching large scale of words
- It should have a dump functionality to let user select what they want to output

Tools

As you can see from the above, this project is not a small one, and it could take months for development and testing. So pick up the right tools is quite important, especially for the extendable request that could happen in future. The following are the main tools we used for developing.

- **Ruby**

Ruby is an easy to learn dynamic open source programming language with a focus on simplicity and productivity. It has an elegant syntax that is natural to read and easy to write. And it is a pure object-oriented programming language with a super-clean syntax that makes programming elegant and fun. Especially the meta-programming technique in ruby makes it even more powerful than other programming languages.

- **Ruby on rails**

Ruby on rails is an open-source framework for developing web-based, database driven applications. Rails takes full advantage of ruby's useful features and make it easier to develop, deploy, and maintain agile web applications.

- **MySQL**

MySQL is an open source database server based on the concept of relational database. It is the most widely used database in creating web applications. The inner structure of MySQL database is some kind of 'fixed' due to its relational database basis. However we can apply some carefully designed schema to satisfy the need of 'dynamic inner structure' for horizontal expansion of lexicon.

We mainly use these three tools in developing Cradle and do our best to make this application easy to use and convenient to extend.

3. System Design and functionality

System design

When we started this project, the most difficult part at beginning was database design. There are two major obstacles to overcome.

- The first one is how to dynamically add properties to every lexicon. As we know that though it is easy to maintain vertically extended lexicon information, it is hard to figure out a way that can dynamically store those horizontally extended information for every lexicon.
- The second one is how to store annotated internal structure and its tagging information of synthetic words in database.

Figure 7.2 shows the design of our system.

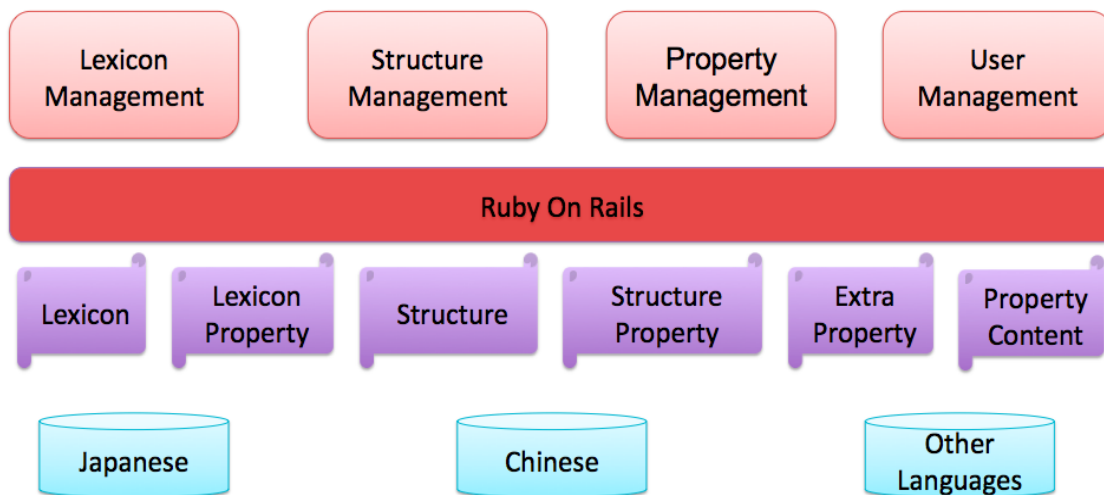


Figure 7.2. Database design of Cradle

As you can see from Figure 7.2, we prepare one database for each language domain, and inside these databases, we use several kinds of tables to hold every

piece of information available. The information held by these tables is described as following.

- lexicon tables: hold basic lexical information like surface, pronunciation, reading etc.
- lexical property tables: hold property information of each lexical entry defined in each dictionary.
- structure tables: hold annotated internal structure of each synthetic word.
- structure property tables: hold annotated tag information for structures of synthetic words.
- extra property tables: hold dynamically created property definition for each lexical entry or structure.
- property content tables: hold the actual tag of dynamically created property for each lexical entry or structure.


Functionalities

With this database design, we develop *Cradle* based on ruby on rails stack, and we realize mainly four kinds of functionalities.

- Lexicon management


This is the main functionality of the whole *Cradle* system, which provides two major actions for users: searching and annotation. We developed quite complicated searching interface (shown in Figure 7.3) that can be used to search every piece of information stored in the system. For example, Figure 7.4 shows the search result of lexical entries that have been annotated with internal structure.

As for annotation, besides the basic edit and save functionality, we also develop specific tagging actions, such as specifying lexicon root or base for Japanese conjugated word. This is necessary because properties like base or root have relationship with other information in the system, and writing their own tagging logic is the only way that can solve this type of problem.



COMPUTATIONAL LINGUISTICS LAB

CRADLE--CHASEN字典管理系统



cradle | [Preference](#) | [User list](#) | [Logout](#)

[日本語辞書](#) | [中文辞典](#)

单词属性

ID:

拼音:

辞典:

更新时间: - :

词类:

单词:

词性:

字数:

状态:

创建者:

更新者:

内部读音:

状态:

更新时间: -

内部成分:

内部词性:

更新者:

结构内部关系:

Figure 7.3. Search interface of Cradle



CRADLE--CHASEN 辞典管理系统

[日本語辞書](#) | [中文辞典](#)

[cradle](#) | [Preference](#) | [User list](#) | [Logout](#)

显示个数: [Review](#) [字典 <=> 显示颜色](#) [显示字段](#) [输出结果](#) include dependency

« Previous | 1 | 2 | ... | 358 | 359 | 360 | 361 | **362** | 363 | 364 | 365 | 366 | ... | 372 | 373 | Next »


条件: 结构状态=CHECKED										11176 Hits
ID	单词	拼音	辞典	词性	状态	词类	结构	构造状态	结构内部关系	
查看	21067	省内外	■	NN	INITIAL	合成词-复合词-合并结构	■	CHECKED	偏正式-MH	
查看	134783	恋爱占卜师	■	NN	NEW	合成词-复合词-分支结构	■	CHECKED	偏正式-MH	
查看	134784	星梦美少女	■	NN	NEW	合成词-复合词-分支结构	■	CHECKED	偏正式-MH	
查看	134785	巴耳麦公式	■	NN	NEW	合成词-复合词-分支结构	■	CHECKED	偏正式-MH	
查看	134786	细菌分类表	■	NN	NEW	合成词-复合词-分支结构	■	CHECKED	偏正式-MH	
查看	134787	突尼斯王国	■	NN	NEW	合成词-复合词-分支结构	■	CHECKED	偏正式-MH	
查看	134788	北陆新干线	■	NN	NEW	合成词-复合词-分支结构	■	CHECKED	偏正式-MH	
查看	134789	露营者日记	■	NN	NEW	合成词-复合词-分支结构	■	CHECKED	偏正式-MH	
查看	134790	西洋陆军棋	■	NN	NEW	合成词-复合词-分支结构	■	CHECKED	偏正式-MH	
查看	134791	人教委员会	■	NN	NEW	合成词-复合词-分支结构	■	CHECKED	偏正式-MH	
查看	134792	高压氧治疗	■	NN	NEW	合成词-复合词-分支结构	■	CHECKED	偏正式-MH	
查看	134793	老人痴呆症	■	NN	NEW	合成词-复合词-分支结构	■	CHECKED	偏正式-MH	
查看	134794	台北市孔庙	■	NN	NEW	合成词-复合词-分支结构	■	CHECKED	偏正式-MH	

Figure 7.4. Search result of Cradle

- Structure management

This functionality is especially developed for tagging internal structure of Chinese and Japanese synthetic words. Usually, synthetic words' structures are described in tree style. We find a way to represent these tree structures in database, and develop a sophisticated but easy to use interface (shown in Figure 7.5) that could let users specify those structures directly in web page and annotate them with different kinds of tags.

And of course, we also make those already defined structures re-useable when defining new structure for other synthetic words, which fits the right common idea of tree style structure of synthetic words (shown in Figure 7.6).



THE HONG KONG POLYTECHNIC UNIVERSITY

CRADLE--CHASEN字典管理系统

cradle | Preference | User list | Logout

日本語辞書

中文辞典

内部构造: 香 港 + 特 - 特 别 区

[Top: 香港特别行政区\(68174\) ==> \[香港\(25906\), 特别行政区0 \]](#)

行政区

变更

特别

变更

选择	更新	ID	单词	拼音	词性	结构	辞典	状态
<input type="radio"/>		23080	特别		AD		■	INITIAL
<input type="radio"/>		23081	特别		JJ		■	INITIAL
<input type="radio"/>		23082	特别		VA		■	INITIAL
Dummy								
	更新	ID	单词	拼音	词性	结构	辞典	状态

V 确定

行政区

变更

选择	更新	ID	单词	拼音	词性	结构	辞典	状态
<input type="radio"/>		26765	行政区		NN	show	■	INITIAL
Dummy								
	更新	ID	单词	拼音	词性	结构	辞典	状态

Figure 7.5. Tree structure annotation interface of synthetic word in Cradle

CRADLE--CHASEN 辞典管理系统

cradle | Preference | User list | Logout

日本語辞書 中文辞典

内部构造

单词详细内容

ID	135339
单词	中华人民共和国香港特别行政区护照
拼音	
词性	NN
辞典	NAIST-ondic-2008
词类	合成词-复合词-分及结构
状态	NEW
备注	
创建者	cradle
更新者	fel-c
更新时间	2010-10-31 22:35:19

[编辑] [删除]

结构详细内容

状态	CHECKED
备注	
更新者	cradle
更新时间	2010-11-14 09:26:40

[编辑] [删除]

内部构造

中华人民共和国香港特别行政区护照

内部结构	中华人民共和国香港特别行政区, 护照
结构内部关系	偏正式-MH

中华人民共和国香港特别行政区

内部结构	中华人民共和国, 香港特别行政区
结构内部关系	偏正式-MH

```

    graph TD
      Root[中华人民共和国香港特别行政区护照] --> Node1[中华人民共和国香港特别行政区]
      Root --> Node2[护照]
      Node1 --> Node3[中华人民共和国]
      Node1 --> Node4[香港特别行政区]
      Node3 --> Node5[中华人民]
      Node3 --> Node6[共和国]
      Node5 --> Node7[中]
      Node5 --> Node8[华]
      Node5 --> Node9[人]
      Node5 --> Node10[民]
      Node6 --> Node11[共]
      Node6 --> Node12[和]
      Node6 --> Node13[国]
      Node4 --> Node14[香港]
      Node4 --> Node15[特别行政区]
      Node15 --> Node16[特别]
      Node15 --> Node17[行政]
      Node15 --> Node18[区]
      Node16 --> Node19[特]
      Node16 --> Node20[别]
      Node17 --> Node21[行]
      Node17 --> Node22[政]
      Node18 --> Node23[区]
    
```

Figure 7.6. Tree structure view of a synthetic word in Cradle

- Property management

This part of management actually has two parts. One part is for those pre-defined properties that have already existed in system, such as part-of-speech, tagging-state, verb-type etc. The other part is for new properties that will be added by the users themselves. As said before, how to let users dynamically create property for each lexical entry in system is a difficult problem in implementation. But with a well-defined database structure, we managed to solve this problem quite successfully. Now, we can dynamically add new properties not only for lexical entries but also for internal structures. And besides, after new properties are added into the system, users can get search functionality on these newly created properties automatically.

- User management

At present, we have only developed a simple user management functionality, which can be used to define three kinds of users: ordinary user, annotator and administrator. Administrators have all the priority of the whole system. Annotators can modify lexicon and structure information, but they have no access to property management. Ordinary users (need login) can only retrieve information from the system, but they have the access to hidden dictionaries that anonymous users (do not need login) have not.

4. Implementation and remaining problems

At present, Cradle is holding 135,767 lexical entries for Chinese and 771,460 for Japanese. We have already used it in tagging the internal structures for both Chinese and Japanese synthetic words. And the performance is quite satisfactory until now.

However, by actually using this system for managing lexical information and doing annotation, we also find that there are several places needing improving:

- After adding several dictionaries and users into system, we find there should be a much more complicated user management functionality than the current one. We plan to rewrite this part to be able to define which user can

access to which dictionary and has what priority on lexicons inside that dictionary.

- Although property can be dynamically added into system, it currently has to be one of these three types: text, category or time. And we find that it is necessary to add more types like integer and real numbers to fit different kinds of information.
- Current system provides functionality to dump search results into text file. However, it does not have upload function to let user create dictionary by uploading a text file.
- Current system uses separated database for each language domain for the intention to improve search speed. However, that does not work as expect in practice. And current database schema for holding horizontal expended information becomes bottleneck of search functionality as dictionary size getting larger.

Chapter 8

Conclusion and Future Work

1. Conclusion

In this dissertation, we try to use machine-learning methods to automatically parse the internal structure information of Chinese synthetic words.

First, we mention the remaining problems in Chinese word segmentation and explain how we would solve them by using Chinese synthetic words analysis. Then we briefly introduce the conceptual definition and categorization of Chinese synthetic words.

After that, we conduct three types of experiments on parsing internal structure information of synthetic words. Because three-character synthetic words are the most typical words with internal structure, we conduct experiments on them using both adjacency model and dependency model.

Next, we describe our annotation work on synthetic words by using the annotation system we create, and propose a tree-based bottom-up parsing approach for general synthetic words by using customized CYK algorithm. Then we apply the proposed approach on all annotated synthetic words. In real experiments, we use features computed from Chinese GigaWord n-gram data and other common statistical features to generate weights for all possible internal parts of words. Then we put these weights into a customized CYK table to predict a best tree for each synthetic word.

In the last part, we describe an extendable lexicon management system we create for language resource management and annotation. After showing the main

functionalities of current system, we briefly introduce the plans of our undergoing development.

As conclusion, we believe our research is the first detailed work specialized on parsing synthetic words' internal structure in Chinese language processing community. We not only proposed a quite complete definition and categorization system for Chinese word and Chinese synthetic word, but also did a lot of surveys and tried various approaches for parsing Chinese synthetic words. And because neither any Chinese synthetic word corpus nor proper tools to create that kind of resources are currently available, we also developed a practical annotation system for tagging the internal structure of synthetic words, which can be used for maintaining large-scale lexicon resources in future.

2. Future work

Chinese synthetic word parsing

As shown in Chapter 6, the final result of internal structure analysis on long synthetic words is not good. After examine the result, we list some reasons that may cause the low accuracy of parsing. Inside of those possible reasons, we believe lacking web-scale n-gram data is the most important one if we want to conduct parsing based on common statistical information. And changing the feature matrix to include more useful features may help the real parser to get better performance too. Therefore, we need to get web-scale Chinese text resources as our analysis base in future. And after getting a good parsing accuracy for internal structure of Chinese synthetic word, we could use that as prior knowledge to recognize the relationship between each internal part pairs, which described in Chapter 2. Finally by using the lexicon management system we create, we can store all these construction information for synthetic words in a proper pattern, which can be used in further research.

Lexicon management system

In order to solve current remain shortages and fit users' needs. The following improvement could be made in future development of Cradle system.

- same database for all language domains
- more sophisticated user management (authentication + authorization)
- more flexible dictionary and feature management
- data import and export ability by user
- more easy to use interface for annotation
- more compatible for other browser rather than firefox
- more clean and readable code

In all of these improvement plans, the most important one is to construct a underlying data schema to realize more flexible dictionary and feature management.

However, due to its relational database characteristic, with the traditional MySQL database server we are using in current Cradle system, it is almost impossible to implement dynamic data schema and ensure searching speed at the same time.

Therefore, searching something new that can guaranty us both dynamic data schema and fast searching speed simultaneously is very important. We suggest that a rather new database backend, ‘MongoDB’ which is a non-relational database server, should be used in future. Because comparing to MySQL, MongoDB has the following fascinating features:

- Average searching speed inside database is faster.
- No need to define data schema beforehand. The underlying data structure can be modified dynamically without re-definition.

In fact, these two features are exactly what we need to improve the fundamental structure of Cradle system. And as a development proposal, all the improvements described above could be implemented by following a new system design shown in Figure 8.1.



Figure 8.1. New system design of Cradle

References

- [1] Masayuki Asahara, K. Fukuoka, A. Azuma, C.L. Goh, Y. Watanabe, Yuji Matsumoto, and T. Tsuzuki. Combination of machine learning methods for optimum chinese word segmentation. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*, pages 134–137, 2005.
- [2] Keh-Jiann Chen and Chao-Jan Chen. Automatic semantic classification for chinese unknown compound nouns. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, 2000.
- [3] Koby Crammer and Yoram Singer. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 2001.
- [4] Yingxian Cui. *XianDai HanYu YuFa XueXi Yu YanJiu RuMen*. Tsinghua University Press, 2004.
- [5] Jenny Rose Finkel and Christopher D. Manning. Nested named entity recognition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 141–150, 2009.
- [6] G. Fu and K.K. Luke. An integrated approach for chinese word segmentation. In *Proceedings of PACLIC 17*, 2003.
- [7] J. Gao, C. Huang, and M. Li. An improved chinese word segmentation system with conditional random field. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 162–165, 2006.
- [8] J. Gao, A. Wu, M. Li, C. Huang, H. Li, X. Xia, and H. Qin. Adaptive chinese word segmentation. In *Proceedings of ACL*, pages 463–470, 2004.
- [9] Chooi-Ling Goh, Masayuki Asahara, and Yuji Matsumoto. Training multi-classifiers for chinese unknown word detection. In *Proceedings of International Conference of Chinese Computing*, 2005.
- [10] Chooi-Ling Goh, Masayuki Asahara, and Yuji Matsumoto. Machine learning-based methods to chinese unknown word detection and pos tag guessing. *Journal of Chinese Language and Computing*, 16(4):185–206, 2006.

- [11] Chooi-Ling Goh, Jia Lu, Masayuki Asahara, and Yuji Matsumoto. A practical morphological analyzer based on penn chinese treebank standard. In *Proceedings of the 12th Annual Meeting of the Association for Natural Language Processing*, pages 540–543, 2006.
- [12] Chooi-Ling Goh, Jia Lu, Yuchang Cheng, Masayuki Asahara, and Yuji Matsumoto. The construction of a dictionary for a two-layer chinese morphological analyzer. In *Proceedings of the 20th Pacific Asia Conference on Language, Information and Computation*, 2006.
- [13] Yuanjian He. Hanyu zhenjia fuheci - cong pubian yufa yuanze kan hanyu fuheci de yuxu, leixing ji jieou. The Chinese University of Hong Kong, 2004.
- [14] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. *A Practical Guide to Support Vector Classification*. Department of Computer Science, National Taiwan University, 2010.
- [15] Chu-Ren Huang, Keh jiann Chen, and Li-Li Chang. Segmentation standard for chinese natural language processing. *International Journal of Computational Linguistics and Chinese Language Processing*, pages 47–62, 1997.
- [16] Chinese Academy of Science Institute of Computing Technology. Cnlp platform. <http://www.nlp.org.cn/>, 2005.
- [17] Wenbin Jiang, Liang Huang, and Qun Liu. Automatic adaptation of annotation standards: Chinese word segmentation and pos tagging: a case study. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language*, 2009.
- [18] Daisuke Kawahara Kun Yu and Sadao Kurohashi. Learning head-modifier pairs to improve lexicalized dependency parsing on a chinese treebank. In *Proceedings of the Sixth International Workshop on Treebanks and Linguistic Theories*, 2007.
- [19] Mark Laue. Corpus statistics meet the noun compound: some empirical results. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, 1995.

-
- [20] Dekang Lin, Kenneth Church, Heng Ji, Satoshi Sekine, David Yarowsky, Shane Bergsma, Kailash Patil, Emily Pitler, Rachel Lathbury, Vikram Rao, Kapil Dalwani, and Sushant Narsale. New tools for web-scale n-grams. In *Proceedings of The International Conference on Language Resources and Evaluation*, 2010.
- [21] Jin Kiat Low, Hwee Tou Ng, and Wenyuan Guo. A maximum entropy approach to chinese word segmentation. In *Proceedings of the 4th SIGHAN Workshop on Chinese Language Processing*, 2005.
- [22] Shengfen Luo and Maosong Sun. Two-character chinese word extraction based on hybrid of internal and contextual measures. In *Proceedings of the 41th Annual Meeting of the Association for Computational Linguistics*, 2003.
- [23] Ryan McDonald, Koby Crammer, and Fernando Pereira. Online large-margin training of dependency parsers. In *Proceedings of ACL*, 2005.
- [24] Ryan Mcdonald and Fernando Pereira. Online learning of approximate dependency parsing algorithms. In *proceedings of the European Chapter of the ACL (EACL)*, 2006.
- [25] Hiroshi Nakagawa, Hiroyuki Kojima, and Akira Maeda. Chinese term extraction from web pages based on compound word productivity. In *Proceedings of the Third SIGHAN Workshop on Chinese Language Processing*, 2004.
- [26] Tetsuji Nakagawa. Chinese and japanese word segmentation using word-level and character-level information. In *COLING '04 Proceedings of the 20th international conference on Computational Linguistics*, 2004.
- [27] Joakim Nivre. *Inductive Dependency Parsing*. Springer, 2006.
- [28] University of Pennsylvania. The penn chinese treebank project. <http://www.cis.upenn.edu/chinese/ctb.html>, 2000.
- [29] Jerome L. Packard. *The Morphology of Chinese-A Linguistic and Cognitive Approach*. Cambridge university press, 2000.

- [30] Emily Pitler, Shane Bergsma, Dekang Lin, and Kenneth Church. Using web-scale n-grams to improve base np parsing performance. In *proceedings of the 23rd International Conference on Computational Linguistics*, pages 889–894, 2010.
- [31] Huihsin Tseng and Keh-Jiann Chen. Design of chinese morphological analyzer. In *Proceedings of the First SIGHAN Workshop on Chinese Language Processing*, 2002.
- [32] Huihsin Tseng, Daniel Jurafsky, and Christopher Manning. Morphological features help pos tagging of unknown words across language varieties. In *Proceedings of the 4th SIGHAN Workshop on Chinese Language Processing*, pages 32–39, 2005.
- [33] Kun Wang, Chengqing Zong, and Keh-Yih Su. Which is more suitable for chinese word segmentation, the generative model or the discriminative one? In *Proceedings of the 23rd Pacific Asia Conference on Language, Information and Computation*, 2009.
- [34] Andi Wu. Customizable segmentation of morphologically derived words in chinese. *International Journal of Computational Linguistics and Chinese Language Processing*, 8(1):1–28, February 2003.
- [35] Emiko Yamada and Yuji Matsumoto. Internal structure representation and analysis of japanese technical terms based on character-wise dependency relation. In *proceedings of Information Processing Society of Japan*, 2009.
- [36] Limin Yao, Mu Li, and Changning Huang. Improving chinese chunking with enriched statistical and morphological knowledge. In *Proceedings of IEEE-Natural Language Processing and Knowledge Engineering*, 2007.
- [37] Hai Zhao. Character-level dependencies in chinese: Usefulness and learning. In *EACL '09 Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, 2009.
- [38] Hai Zhao and Chunyu Kit. Incorporating global information into supervised learning for chinese word segmentation. In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*, pages 66–74, 2007.

- [39] Hai Zhao and Chunyu Kit. Unsupervised segmentation helps supervised learning of character tagging for word segmentation and named entity recognition. In *Proceedings of The 6th SIGHAN Workshop on Chinese Language Processing*, 2008.
- [40] Guodong Zhou. A chunking strategy towards unknown word detection in chinese word segmentation. In *Proceedings of the 2nd International Joint Conference on Natural Language Processing*, pages 530–541, 2005.
- [41] Qiang Zhou. Automatic rule acquisition for chinese intra-chunk relations. In *Proceedings of the 3rd International Joint Conference on Natural Language Processing*, 2008.

Appendix

A. POS tagset in ChaSen Chinese Dictionary

POS Tag	Description	Examples
AD	adverb	还
AS	aspect marker	着
BA	把 in ba-construction	把, 将
CC	coordinating conjunction	和
CD-NOR	cardinal number	一, 百
CD-AFF	affix used in cardinal number	点, 多
CS	subordinating conjunction	虽然
DEC	的 in a relative-clause	的
DEG	associative 的	的
DER	得 in V-de const. and V-de-R	得
DEV	地 before VP	地
DT	determiner	这
ETC	for words 等, 等等	等, 等等
FW	foreign words	a, z, A, Z
IJ	interjection	啊
JJ	other noun-modifier	男, 共同
LB	被 in long bei-const	被, 给
LC	localizer	里
M	measure word	个
MSP	other particle	所
NN	common noun	书
NR-PER-FAM	CJK family name	吴, 松本
NR-PER-GIV	CJK given name	翠玲, 樱子
NR-PER-FOR	transliteration (foreign) person name	阿里巴巴
NR-PER-OTH	other person name	关公, 鲁迅

POS Tag	Description	Examples
NR-LOC	place name	中国, 富士山
NR-ORG	organization name	富士通, 民主党
NR-OTH	other proper name	木星, 秦朝
NT-AFF	affix used in temporal noun	年, 月
NT-NOR	temporal noun	今天, 冬季
OD-AFF	affix used in ordinal number	第
OD-NOR	ordinal number	首, 初
ON	onomatopoeia	哈哈, 哗哗
P	preposition excl. 被 and 把	从, 对于
PN	pronoun	他, 大家
PU	punctuation	?。、
SB	被 in short bei-const	被, 给
SP	sentence-final particle	吗, 呢
VA	predicative adjective	红, 雪白
VC	是	是
VE	有 as the main verb	有
VV	other verb	走

List of Publications

Journal Papers

- Jia Lu, Masayuki Asahara and Yuji Matsumoto. "Annotation and Classification of Three-Character Chinese Synthetic Words". The International Journal of Computer Processing of Languages, Vol.21, No.2, 101-122. 2008.

International Conference/Workshop Papers

- Jia Lu, Masayuki Asahara and Yuji Matsumoto. "Analyzing Chinese Synthetic Words with Tree-based Information and A Survey on Chinese Morphologically Derived Words". In Proceedings of the sixth SIGHAN Workshop on Chinese Language Processing, 53-60. 2008.

Other Publications

- Jia Lu, Masayuki Asahara and Yuji Matsumoto. "Cradle: An Extendable Lexicon Annotation and Management System". In Proceedings of Public Workshop of Grant-in-Aid for Scientific Research in Priority Areas JAPANESE CORPUS, 111-116. 2010.
- Jia Lu, Masayuki Asahara and Yuji Matsumoto. "Chinese Synthetic Word Analysis with Tree-based Structure Information". In Proceedings of the 13th Annual Meeting of the Association for Natural Language Processing, 827-830. 2007.
- Chooi-Ling Goh, Yuchang Cheng, Jia Lu, Masayuki Asahara and Yuji Matsumoto. "A Practical Morphological Analyzer Based on Penn Chinese Treebank Standard". In Proceedings of the 12th Annual Meeting of the Association for Natural Language Processing, 540-543. 2006.
- Chooi-Ling Goh, Jia Lu, Yuchang Cheng, Masayuki Asahara and Yuji Matsumoto. "The Construction of a Dictionary for a Two-layer Chinese Morphological Analyzer". In Proceedings of the 20th Pacific Conference on Language, Information and Computation. Vol.20, 332-340. 2006.