Computational Linguistics and Chinese Language Processing
Vol. 21, No. 1, June 2016, pp. 1-28

A Segmentation Matrix Method for Chinese Segmentation Ambiguity Analysis

Yanping Chen[*,+], Qinghua Zheng[+], Feng Tian[+], and Deli Zheng[+]

Abstract

Chinese Segmentation Ambiguity (CSA) is a fundamental problem confronted when processing Chinese language, where a sentence can generate more than one segmentation paths. Two techniques are commonly used to identify CSA: Omni-segmentation and Bi-directional Maximum Matching (BiMM). Due to the high computational complexity, Omni-segmentation is difficult to be applied for big data. BiMM is easier to be implemented and has a higher speed. However, recall of BiMM is much lower. In this paper, a Segmentation Matrix (SM) method is presented, which encodes each sentence as a matrix, then maps string operation into set operations. To identify CSA, instead of scanning a whole sentence, only specific areas of the matrix are checked. SM has a computational complexity close to BiMM with recall the same as Omni-segmentation. In addition to CSA identification, SM also supports lexicon-based Chinese word segmentation. In our experiments, based on SM, several issues about CSA are explored. The result shows that SM is useful for CSA analysis.

Keywords: Segmentation Matrix, Segmentation Ambiguity.

1. Introduction

Chinese characters are originated from hieroglyphic and written next to each other without delimiter in between. The lack of orthographic words makes Chinese word segmentation difficult. It is often that a Chinese sentence can be parsed into several segmentation paths, which results in the Chinese Segmentation Ambiguity (CSA) problem. It can be roughly classified into two categories: Overlapping Ambiguity (OA) and Combinational Ambiguity (CA)[1] (Liang, 1984; Sun, 2001). For the OA problem, a sentence contains at least two

[*] Guizhou University

[+] Xi'an Jiaotong University
E-mail: ypench@gmail.com; qhzheng@mail.xjtu.edu.cn; ftian@sei.xjtu.edu.cn,;
delizheng.2009@gmail.com

[1] Formal definitions of CA and OA are given in Section 2.

overlapped words. For example, “温柔和” contains two overlapped words: “温柔” (Gentle) and “柔和” (Soft). The character “柔” can be assembled with either “温” and “和”. Only one is suitable in a given context. In Chinese, every character can be either a morpheme or a word (Li, 2011). Then, given a word containing more than one characters, whether it is appropriate to segment it will lead to the CA problem. For example, “温柔” (Gentle) can be further segmented into “温/柔” (Warm/ Soft).

In Chinese, the OA is prevalent. For example, in the Penn Chinese Treebank corpus, there are 39% sentences are identified with this ambiguity. Therefore, the OA is widely studied in this field. When the CA is under consideration, the problem is more serious. For example, in the lexicon of our experiments, 75.25% words have the CA problem, even using a loose definition (See Definition 4 of Section 3). Furthermore, CA and OA are not independent. They often co-occur within a sentence, which worsens the performance of Chinese word segmentation (Chen *et al*., 2012). The problem is deteriorated by the fact that Chinese has a large number of characters and words[2].

To identify CSA, two techniques are commonly used: Omni-segmentation and BiMM. Omni-segmentation tries to traverse every segmentation path in a sentence. All ambiguities can be identified. The problem of Omni-segmentation is that it has the highest computational and space complexities. For example, a sentence “江泽民在北京人民大会堂会见参加全国法院工作会议和全国法院系统打击经济犯罪先进集体表彰大会代表时要求大家要充分认识打击经济犯罪工作的艰巨性和长期性” (Meanings of this sentence can be ignored[3]) may generate 3,764,387,840 segmentation paths (Wang *et al*., 2004). When a large-scale dataset is confronted, this method is difficult to be applied, unless additional information is available, *e.g*., statistic information (Wang *et al*., 2009). BiMM is frequently adopted for identifying CSA (Li *et al*., 2003). It is easier to be implemented and has a higher speed. The disadvantage of BiMM is that overlapping ambiguity strings with even length (counted by characters) cannot be identified[4] (Sun *et al*., 2001; Chang *et al*., 2008). Furthermore, BiMM only identifies MOAS[5]. Without addition information, it cannot find individual Overlapping Ambiguity String (OAS) in a sentence. Therefore, many studies are mainly focused on MOAS

[2] Currently, more than 13000 characters and 69000 words are used by native Chinese people (*http://www.cp.com.cn/*).

[3] In Beijing's Great Hall, when meeting representatives attending the national court and the national court system against economic crime on behalf of advanced collective awards ceremony, Zemin Jiang asks everyone to fully understand that the work of combating economic crime is arduous and long-term.

[4] Section 2.2 gives an example of this combination pattern.

[5] A MOAS is an ambiguity string that no overlapping ambiguity is detected on both sides of the string. Formal definition can be seen in Definition 7.

(Sun *et al*., 1999; Li *et al*., 2008; Qiao *et al*., 2008; Li, 2011).

In this article, a Segmentation Matrix (SM) method is presented. It encodes lexical information of a sentence as a matrix. Then, set theory is developed to analyze CSA. With the complexity closing to BiMM, SM can identify every type of CSA the same as Omni-segmentation. In addition to CSA identification, SM is also available for Chinese word segmentation. Several lexicon-based methods are fully supported. Making use of the SM method, in our experiments, characteristics of CSA are studied, which show informative conclusions of CSA.

The contribution of this paper includes,

1. A SM approach is proposed, which encodes lexical information in a structured form. SM can make better use of sentence structure information for CSA analysis.
2. Formal definitions about CSA are defined under the framework of set theory, which maps string operations into set operations reducing the computational complexity.
3. Based on SM, characteristics of CSA are investigated. And several issues about CSA are studies.

The remainder of this paper is structured as follows: Section 2 reviews previous works. Section 3 gives formal definitions and notations about CSA. The SM method is discussed in Section 4. In Section 5, several issues about CSA are analyzed. The conclusion is given in Section 6.

2. Related Work

Given a sentence, CSA is identified when more than one segmentation paths are found. Therefore, CSA identification and Chinese word segmentation are two aspects of a problem. In this section, we first give a simple overview about Chinese word segmentation methods. Then CSA identification approaches are discussed.

2.1 Chinese Word Segmentation

Chinese word segmentation methods can be roughly classified into three categories: lexicon-based methods, statistical-based methods and hybrid methods.

Lexicon-based methods are easy to be implemented and has a high speed. They are still used for Chinese word segmentation in many applications. Maximum Matching (MM) is the most popular lexicon-based method. It is a greedy algorithm implemented by scanning a sentence from one side to another and greedily matching the longest lexicon entry until the end of a sentence is reached. There are two MM methods: Forward Maximum Matching (FMM) and Backward Maximum Matching (BMM). FMM scans from left to right, and BMM starts from the opposite direction.

In statistical-based methods, word segmentation is the way to find a segmentation path which has the maximum probability. They take advantages of mathematical models, such as Naive Bayesian (Li *et al*., 2003), Hidden Markov Model (Zhang *et al*., 2003), Conditional Random Fields (Peng *et al*., 2004; Tseng *et al*., 2005), Maximum Entropy (Xue, 2003), Graph-based Model (Zeng *et al*., 2013) and Compression-based algorithm (Teahan *et al*., 2000). There are research combine generative model with discriminative model, *e.g*., Wang *et al*. (2012). According the type of segmentation units, a sentence can be treated as a character sequence (character-based model) or a word sequence (word-based model). Studies were shown that the character-based approach are more successful (Xue, 2003; Zhao *et al*., 2010).

Hybrid methods integrate statistical-based and the lexicon-based methods (Gao *et al*., 2005) or utilize a joint model combining word segmentation with POS tagging or parsing (Wang *et al*., 2013; Sun, 2011; Li *et al*., 2011; Hatori *et al*., 2012). Hybrid methods try to use syntactic, semantic analyses or external knowledge to improve the performance (Wu *et al*., 1998; Huang *et al*., 2007; Zeng *et al*., 2011; Wu *et al*., 2011; Christiansen *et al*., 2011).

In statistical-based and hybrid methods, the tasks of word segmentation and CSA identification are combined into a unified framework. Therefore, the CSA problem is not obviously considered. As for the lexicon-based methods, greedy algorithms are used, which results in the CSA problem.

2.2 Chinese Segmentation Ambiguity Identification

In order to discuss related work, in this section, we use the same example provided by Wang *et al*. (2004): “当原子结合成分子时”[6]. The right segmentation path should be “当/原子/结合/成/分子/时” (when/ atoms/ combine/ molecules/ the time). The overlapping ambiguous string is “结合成分子时”. It can also be segmented into “结合/成分/子时” (combine/ ingredient/ midnight).

The central issue of CSA identification is that trying to find all possible segmentation paths. BiMM is the most popular method for CSA identification (Gao *et al*., 2011; Yao *et al*., 2012). It is implemented by running FMM and BMM respectively. The two outputs of FMM and BMM are compared. Different outputs imply the existence of segmentation ambiguity. The main disadvantage of BiMM is that overlapping ambiguity strings with even length cannot be identified. In this situation, both of FMM and BMM have the same output. As shown in Figure 1, two outputs of BiMM are the same.

[6] It can be translated into: “when atoms combine into molecules”.

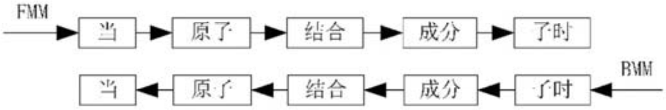


Figure 1. BiMM Method

Omni-segmentation tries to find every segmentation path in a sentence, which can be illustrated by a tree structure as shown in Figure 2. The root represents the start of a sentence. Nodes represent words of a sentence. Each branch implies a possible segmentation path.

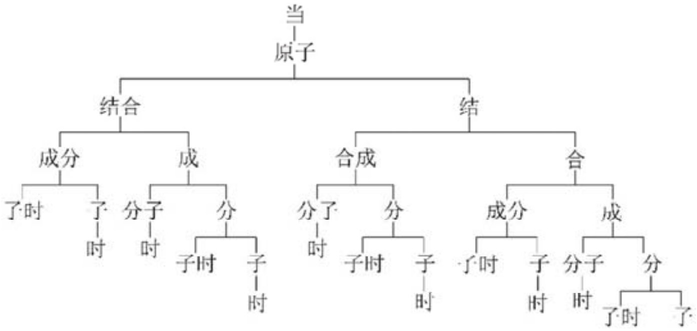


Figure 2. Omni-segmentation Method

The Directed Acyclic Graph (DAG) method was proposed by Zhang *et al*. (2002) as Figure 3 shows. Given a sentence represented as $c_0 \cdots c_{n-1}$ (c_i denotes Chinese characters), vertices $v_0, \cdots, v_n$ are used to separate it. Then, $v_0 c_0 v_1 c_1 \cdots v_{n-1} c_{n-1} v_n$ is generated. v_0 and v_n are the start and the end vertexes. If a substring $c_i \cdots c_j$ $(0 \leq i, j \leq n-1)$ matches a lexicon entry, then a directed edge (v_i, v_{j+1}) is added.

The DAG is used to collect possible segmentation paths. According to Zhang *et al*. (2002), if the 8-shortest paths are collected, this method can receive performance about 99.90% in recall to find correct segmentation paths.

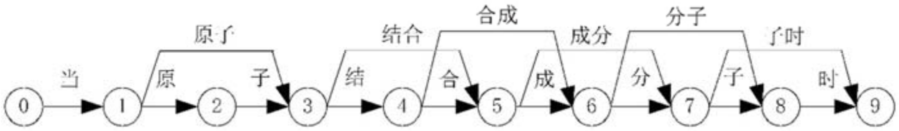


Figure 3. Directed Acyclic Graph

Wang *et al*. (2004) proposed a Maximum No-cover Ambiguity Graph (MNAG) as Figure 4 shows. Based on the principle of *Choosing the Longer Word*, MNAG can identify all overlapping ambiguities. This approach can reduce the number of segmentation paths. But identification of the combinational ambiguity is ignored.

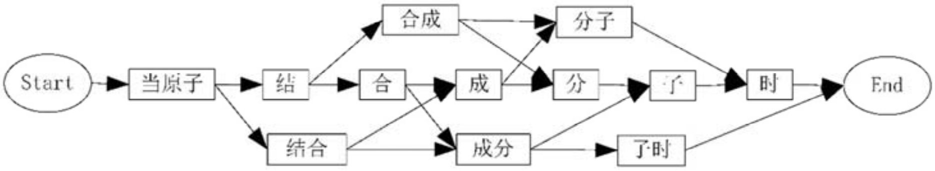


Figure 4. Maximum No-cover Ambiguity Graph

The word lattice method was proposed for Chinese word segmentation (Jiang *et al*., 2008). It is built by merging the output of outer segmenters. This method is mainly used as a re-ranking strategy. As shown in Figure 5, lattice nodes denote positions between characters. Edges covering subsequences of sentence denote words (Wang *et al*., 2013).

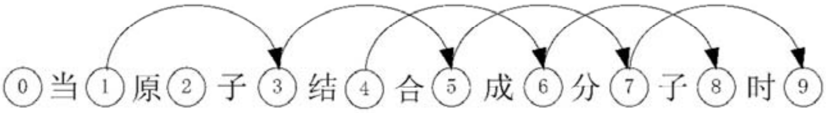


Figure 5. Word Lattice

There are other approaches proposed for CSA analyses, such as the *overlapping ambiguity elimination* model (Yao *et al*., 2012), the word-by-word scanning based maximum matching algorithm (Zhang *et al*., 2006; Sun *et al*., 2009), the method based on type theory (Gao *et al*., 2009) and the *coupling degree of double characters* method (Wand *et al*., 2007). These methods mainly focus on the overlapping ambiguity. Problems of combinational ambiguity and the difference between MOAS and OAS are rarely studied. In this paper, we propose a SM method. The detail of SM is discussed in Section 4. In the following, we first introduce definitions and notations used in this paper.

3. Definitions and Notations

Let $\mathbf{M}$ to be a segmentation matrix, $\mathbf{M}(i,j)$ is an element of $\mathbf{M}$ with coordinates Row i and Column j. A sentence is referred as S. The length of S is supposed to be n. We define two sets as:

$$\mathbf{P} = \{\ i \mid \text{an index of character position in } S; 0 \leq i \leq n-1; i \in N\}$$[7]

$$\mathbf{W} = \{w \mid w \text{is a lexicon entry}\}$$

where N are the natural numbers. $\mathbf{P}$ is a partial order set. $\mathbf{W}$ denotes an employed lexicon. A closed interval $[i,j] = \{x | x \in \mathbf{P}, i \leq x \leq j\}$ denotes subset of $\mathbf{P}$. $S[i,j]$ $(i,j \in \mathbf{P}, i \leq j)$ represents substring of S starting from the ith character to the jth character. By means of set operations, sentence operations are defined as follows

[7] Note that: all character positions are indexed from 0 to $n-1$.

$$(S[i,j]\cup S[i',j'])=S([i,j]\cup[i',j'])$$
$$(S[i,j]\cap S[i',j'])=S([i,j]\cap[i',j'])$$
$$S[i,j]\subseteq S[i',j'] \text{ iff } [i,j]\subseteq[i',j']$$
$$S[i,j]=S[i',j'] \text{ iff } [i,j]=[i',j']$$

Note that all sentence operations are implemented within indexes belonging to the same sentence S. Otherwise, these operations are nonsense.

Based on sentence operations, formal definitions about *segmentation path*, *combinational ambiguity*, *overlapping ambiguity*, etc. are defined as follows.

Definition 1: Let $\{[i,j_1],[j_1+1,j_2],\cdots,[j_m+1,j]\}$ to be a partition of $[i,j]$, then $\{S[i,j_1],S[j_1+1,j_2],\cdots,S[j_m+1,j]\}$ is a **Segmentation Path** of $S[i,j]$, referred to as $S(j_1,j_2,\cdots,j)$ or $S[i,j_1]/S[j_1+1,j_2]/\cdots/S[j_m+1,j]$.

In other words, a segmentation path is a partition of $\mathbf{P}$. For example, let S=“温柔和善解人意” (gentle and understanding), because $\{[0,1],[2,2],[3,6]\}$ is a partition of $[0,6]$, then $S(1,2,6)$ is a segmentation path (or $S[0,1]/S[2,2]/S[3,6]$), which denotes “温柔/和/善解人意” (gentle/ and/ understanding). In this paper, $S[i,j]$ (square bracket) represents substring of S, and $S(i,j)$ (parentheses) represents a segmentation path.

Definition 2: Let $SP=\{[i,j_1],[j_1+1,j_2],\cdots,[j_m+1,j]\}$ to be a segmentation path, if $\forall w\in SP$ such that $w\in\mathbf{W}$. Then, the segmentation path SP is **in accord with $\mathbf{W}$**.

In this paper, we assume that all mentioned segmentation paths satisfy this constraint.

Definition 3: Let $S[i,j]\in\mathbf{W}$. If $SP=\{[i,j_1],[j_1+1,j_2],\cdots,[j_m+1,j]\}$ is a partition of $[i,j]$, then S has the **Combinational Ambiguity** (CA), and $S[i,j]$ is a Combinational Ambiguity String (CAS).

For example, let S=“温柔和善解人意”, $S[3,6]$=“善解人意”, $S[3,6]\in\mathbf{W}$ and $\{[3,3],[4,4],[5,6]\}$ is a partition of $[3,6]$. Because $S[3,3]$=“善”, $S[3,3]\in\mathbf{W}$, $S[4,4]$=“解”, $S[4,4]\in\mathbf{W}$, $S[5,6]$=“人意”, $S[5,6]\in\mathbf{W}$, therefore, S has combinational ambiguity. $S[3,6]$ is a CAS, referred to as $CAS[3,6]$.

In Chinese, almost every character can function either as a word or as a morpheme (Chen *et al*., 1998). If Definition 3 is adopted, then words exceeding two characters will lead to the combinational ambiguity. Because disambiguation for combinational ambiguity is difficult (Luo *et al.*, 2002). Therefore, to reduce the combinational ambiguity problem, the following combinational ambiguity definition in a narrow sense is proposed.

Definition 4: Let $S[i,j] \in \mathbf{W}$, if $\exists j'(i \le j' < j)$ such that $S[i,j'] \in \mathbf{W} \wedge S[j'+1,j] \in \mathbf{W}$, then S has the **Narrow Sense Combinational Ambiguity**.

This definition is the same as that proposed by Liang (1984). In this paper, the narrow sense combinational ambiguity is used as our default definition, also referred as combinational ambiguity except where noted.

Definition 5: Let $S[i,j] \in \mathbf{W} \text{ and } S[i',j'] \in \mathbf{W}$, if $i < i' \le j$ and $[i,j] \cap [i',j'] \ne \varnothing$, then S has the **Overlapping Ambiguity** (OA). $S[i,j] \cup S[i',j']$ is an Overlapping Ambiguity String (OAS).

If $S[i,j]$ and $S[i',j']$ have overlapping ambiguity, then $S[i,j] \cap S[i',j']$ is an **Overlapping Chain String** (OCS) and $|S[i,j] \cap S[i',j']|$ is **Overlapping Chain Length** (OCL), where $|S[i,j] \cap S[i',j']|$ is the cardinality of $S[i,j] \cap S[i',j']$.

For example, let S =“温柔和善解人意”, $S[0,2]$=“温柔和”, $S[0,1] \in \mathbf{W}$ and $S[1,2] \in \mathbf{W}$. Because $[0,1] \cap [1,2] = [1,1] \ne \varnothing$, so that S has overlapping ambiguity. $S[0,1] \cup S[1,2] = S[0,2]$ is an OAS. $S[0,1] \cap S[1,2] = S[1,1]$ is an OCS, and OCL of $S[0,2]$ is 1.

In this paper, CAS and OAS are collectively referred as **Ambiguity String** (AS). Prefixes OA, CA and OC are used to indicate properties of $S[i,j]$ (Overlapping Ambiguity, Combinational Ambiguity and Overlapping Chain). For example, $OAS[i,j]$ means that $S[i,j]$ is an OAS.

Definition 6: Given $OAS[i,j]$ and $OAS[i',j']$, if $OAS[i,j] \cap OAS[i',j'] \ne \varnothing$, then $OAS[i,j]$ and $OAS[i',j']$ are **Addable**.

If $OAS[i,j]$ and $OAS[i',j']$ are addable, then the sum of $OAS[i,j]$ and $OAS[i',j']$ is $S([i,j] \cup [i',j'])$. It is also an OAS. If two OAS are addable, the overlapping chain string of $OAS([i,j] \cup [i',j'])$ can be calculated by

$$f(OAS[i,j] \cup OAS[i',j']) = f(OAS[i,j]) \cup (\overline{f(OAS[i,j])} \cap \overline{f(OAS[i',j'])}) \cup f(OAS[i',j']) \quad (1)$$

where $\overline{f(OAS[i,j])}$ is the relative complement of $f(OAS[i,j])$ in $OAS[i,j] \cup OAS[i',j']$.

In this field, the Maximum Overlapping Ambiguity String (MOAS) is widely used. It is defined as follows.

Definition 7: In a given sentence S, if an $OAS[i,j]$ is not addable with other OAS in S, then this $OAS[i,j]$ is a **Maximum Overlapping Ambiguity String** (MOAS).

For example, S_1=“温柔和善解人意” has three OAS: $OAS[0,2]$, $OAS[1,3]$ and $OAS[2,6]$. All of them are addable. The result is $MOAS[0,6]$. By Eq. (1), an overlapping chain string of S_1 is $f(S[0,6]) = OCS[1,3]$. For another example, S_2=“逐渐变成暗红色” has $OAS[0,3]$ and $OAS[4,6]$. $OAS[0,3]$ and $OAS[4,6]$ are not addable, then both of them are MOAS.

Given a set of OAS (referred as $\mathcal{A}_{OAS}$) in a sentence, the set of MOAS (referred to as $\mathcal{A}_{MOAS}$) is computed by merging every OAS that are *addable*. It is computed as follows.

$$\mathcal{A}_{MOAS} = \{a | (s, t \in (\mathcal{A}_{OAS} \cup \mathcal{A}_{MOAS})) \wedge (a = s \cup t) \wedge (s \cap t \neq \varnothing)\} \quad (2)$$

Because $\mathcal{A}_{MOAS}$ appears on both sides of this equation, it is an iterative process. It will be discussed in Section 4.2.3 that $\mathcal{A}_{MOAS}$ is easy to be implemented, because all elements in $\mathcal{A}_{OAS}$ and $\mathcal{A}_{MOAS}$ are ordered.

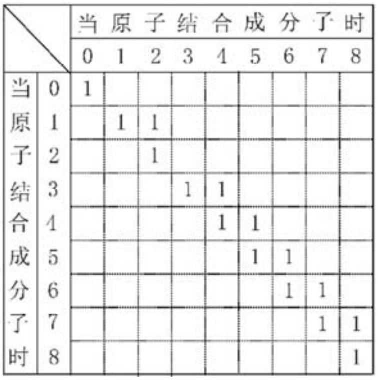
In ancient Chinese, there are no punctuations between sentences. Currently, symbols such as the period (“。”), question mark (“？”), exclamatory mark (“！”), semicolon (“；”) or comma (“，”) are widely used as sentence boundaries. The problem is that using of the comma is ambiguous. It may function as a sentence boundary or a separation of clauses. Therefore, disambiguation of sentence boundary is required (Xue *et al*., 2011). Because lots of language characteristics cannot exist while crossing punctuation, *e.g*., segmentation ambiguity, named entity, etc. (Chen *et al*., 2015b), **Sentence Fragment** is used to denote *substring of a sentence divided by punctuations.*

Definition 8: Sentence fragment is a substring of a sentence that contains no punctuation.

This notion is useful for Chinese NLP, *e.g*., Zhang *et al*. (2013), especially for unsupervised machine learning method, *e.g*., Zhang *et al*. (2003), Li *et al*. (2009).

4. Segmentation Matrix

Figure 6 gives an example of SM. Coordinates of SM represent characters of S. The element data type of SM is Boolean. $\mathbf{M}(i,j)=1$ means that word $S[i,j]\in\mathbf{W}$. To build SM of a sentence, by scanning a given sentence, when a lexicon entry is matched, the corresponding element is set to 1, otherwise to 0.



		当	原	子	结	合	成	分	了	时
		0	1	2	3	4	5	6	7	8
当	0	1								
原	1		1	1						
子	2			1						
结	3				1	1				
合	4					1	1			
成	5						1	1		
分	6							1	1	
了	7								1	1
时	8									1

Figure 6. Segmentation Matrix

4.1 Ambiguity Identification on SM

Following the definition of the *Combinational Ambiguity in a Narrow Sense* (See Definition 4), Solution 1 is proposed to identify combinational ambiguity strings.

Solution 1: For $\forall i\forall j(\mathbf{M}(i,j)=1)$, if $\exists j'(i\leq j'<j\wedge\mathbf{M}(i,j')=1\wedge\mathbf{M}(j'+1,j)=1)$, then $S[i,j]$ is a CAS.

For example, in Figure 6, $\mathbf{M}(i,j)=\mathbf{M}(1,2)=1$, $\mathbf{M}(i,j')=\mathbf{M}(1,1)=1$, $i=1$, $j=2$, $j'=1$, $\mathbf{M}(j'+1,j)=\mathbf{M}(2,2)=1$, then $S[1,2]$ is a CAS. Based on Solution 1, Algorithm 1 gives the implementation of this solution[8].

[8] In this paper, the algorithms are given in C++ codes. Some changes are made for the sake of simplicity and convenience.

Algorithm 1

CA Identification on SM

Input:

1. SegMatrix[][], SM;

2. n, length of the given sentence;

3. l, length of the longest lexicon entry;

Output:

Combinational ambiguity strings are stored in the vector CAS_v;

Method:

```
(0)  for(int $i = 0$; $i < n$; $i + +$){
(1)      for(int $j = min(n - 1, i + l)$; $j > i$; $j - -$){
(2)          if(SegMatrix[i][j]){                    \\$\mathbf{M}(i, j) = 1$
(3)              for(int $j' = j - 1$; $j' \geq i$; $j' - -$){
(4)                  if(SegMatrix[i][j'] && SegMatrix[$j' + 1$][j]){
(5)                      $CAS_v$.push_back(i, j', j);
(6)  }   }   }   }   }
```

As Algorithm 1 shows, three loops are used. Except the outer loop has an index n, another two nested loops have cycle indexes less than l. Therefore, the complexity of combinational ambiguity identification is $O(l^2 n)$. The function min in Row (1) is adopted to decrease the search space. If a “break” clause is in the pace after Row (5), the algorithm will return when the combinational ambiguity is identified. Otherwise, as it shows, every combinational ambiguity in the sentence is collected.

Following Definition 3, Solution 2 is proposed to identify the overlapping ambiguity string.

Solution 2: For $\forall i \forall j (M(i, j) = 1)$, if $\exists i' \exists j' (\mathbf{M}(i', j') = 1 \land i' < i \land j' < j \land i \leq j')$, then $S[i', j]$ is an OAS, $S[j', i]$ is the overlapping chain string, and the overlapping chain length is $j' - i + 1$.

For example, in Figure 6, because $\mathbf{M}(i, j) = \mathbf{M}(4, 5) = 1$, $\mathbf{M}(i', j') = \mathbf{M}(3, 4) = 1$, $i = 4$, $j = 5$, $i' = 3$ and $j' = 4$, then $S[3, 5]$ is an OAS, overlapping chain length equals 1 ($j' - i + 1 = 4 - 4 + 1 = 1$). Algorithm 2 implements the Solution 2.

Algorithm 2

Overlapping Ambiguity Identification on SM

Input:

1. SegMatrix[][], SM;
2. n, length of the given sentence;
3. l, length of the longest lexicon entry;

Output:

Overlapping ambiguity is stored in vector OAS_v;

Method:

```
for(int $i = 0$; $i < n$; $i++$){
    for(int $j = min(n-1, i+l)$; $j > i$ && $i < n$; $j--$){
        if(SegMatrix[i][j]){                              \\$\mathbf{M}(i,j) = 1$
            for(int $i' = i - 1$; $i' \geq 0$; $i'--$){
                for(int $j' = j - 1$; $j' \geq i$; $j'--$){
                    if(SegMatrix[i'][j'])\{               \\$\exists i' \exists j' (\mathbf{M}(i',j') = 1)$
                        $OAS_v$.push_back(i, j, i', j');
}   }   }   }   }   }
```

In Algorithm 2, four loops are used. The outer loop has index n. The other three nested loops have cycle index less than l. The complexity of overlapping ambiguity identification is $O(l^3 n)$. Using Algorithm 2, instead of MOAS, each overlapping ambiguity string is recognized individually. After every OAS is identified, MOAS can be obtained by merging OAS that is addable (See Eq. (2)).

4.2 Segmentation on SM

In this section, we illustrate how Chinese word segmentation algorithms can be implemented on SM. Four lexicon based methods (FMM, BMM, BiMM and Omni-segmentation) are discussed. By mapping string operations into set operations, these processes only implement Boolean operations, which reduces the computing complexity.

4.2.1 FMM

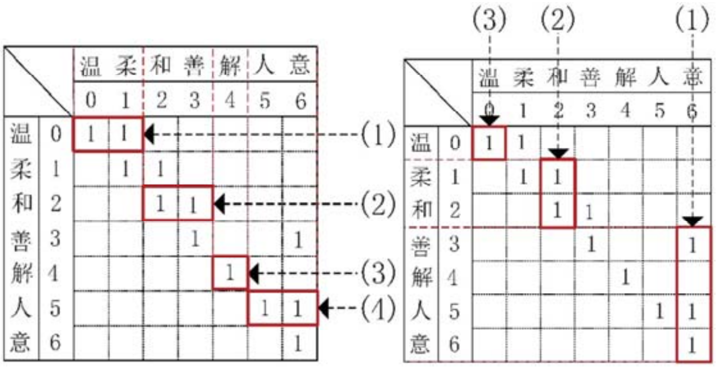
FMM is implemented by scanning each row of SM from right to left. If $\mathbf{M}(i,j)$ equals 1, then hold the coordinate j as j_0, and scan from the (j_0+1)th row again. Iterate this way until the end of the SM ($j = n-1$) is reached. Then, $S(j_0, j_1, \cdots, n-1)$ is a segmentation path of FMM. Step 1 to 3 give an example of this algorithm. Figure 7(a) shows the visualized process.

Step 1: Scan the 0th row from Column 6 to Column 0. Hit 1 at $\mathbf{M}(0,1)$, then set 1 as j_0.

Step 2: Scan the (j_0+1)th row in the same way, if $\mathbf{M}(i,j)$ equals 1, record every j.

Step 3: Iterate this way until the column-coordinate j equals 6.

As shown in Figure 7(a), the output is $S(1,3,4,6)$.



		温	柔	和	善	解	人	意
		0	1	2	3	4	5	6
温	0	1	1					
柔	1		1	1				
和	2			1	1			
善	3				1			1
解	4					1		
人	5						1	1
意	6							1

(a) FMM

		温	柔	和	善	解	人	意
		0	1	2	3	4	5	6
温	0	1	1					
柔	1		1	1				
和	2			1	1			
善	3				1			1
解	4					1		
人	5						1	1
意	6							1

(b) BMM

Figure 7. Maximum Matching Segmentation

4.2.2 BMM

BMM is similar as FMM. The difference is that BMM processes the last column first and scan each column from top to bottom. If $\mathbf{M}(i,j)$ equals 1, hold j and restart from $(i-1)$th column again until the 0th column is reached.

For example, in Figure 7(b), first hit 1 at $\mathbf{M}(3,6)$, then hold 6 and scan from the 2th column again. In Column 2, $\mathbf{M}(1,2)$ equals 1, restart from the $(i-1)$th column until the column-coordination 0 is met. The output of BMM in this example is $S(0,2,6)$.

4.2.3 BiMM on SM

BiMM is implemented by running both FMM and BMM respectively. Two outputs are compared. Let $S_F(\cdots,i,m,\cdots,k,j,n-1)$ and $S_B(\cdots,i,s,\cdots,t,j,n-1)$ to be the output of FMM and BMM. Comparing S_F and S_B from right to left, if both coordinates[9] have the same value, hold this same value and decrease both by 1. Then, compare the new

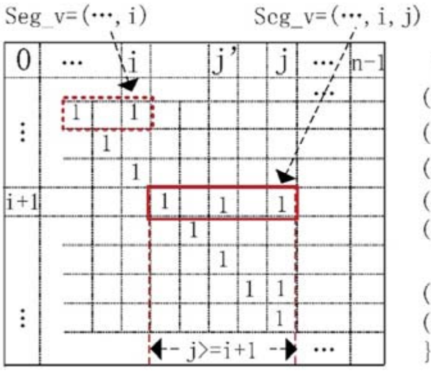
[9] In this place, segmentation path $S(j_1,j_2\cdots,j)$ are seen as a vector. The values of $S(j_1,j_2\cdots,j)$ are j_1, j_2, etc. The coordinates of $S(j_1,j_2\cdots,j)$ are the position index of this vector. For example, in $S(j_1,j_2\cdots,j)$, the value of coordinate 0 is j_1.

coordinates again and always update the held value if both are equal, until the unequal value is met for the first time. Now, the held value is the end of OAS (*e.g.* j in S_F and S_B). Subtract 1 to the coordinate with larger value. Continue this way, until the equal value is found again (*e.g.* i in S_F and S_B). At last, it is the start of the OAS. In this example, the OAS is $S[i+1, j]$. Iterate this way until both S_F and S_B are traversed.

4.2.4 Omni-segmentation on SM

Omni-segmentation tries to find every segmentation path in a given sentence. The number of segmentation paths may explode tremendously. It has the highest computational and space complexity. Based on SM, utilizing segmentation ambiguity information, we can apply Omni-segmentation method on substrings that have the segmentation ambiguity problem, then reducing generated segmentation paths.

For the reason that Omni-segmentation is useful in Chinese NLP, *e.g.*, Chen *et al.* (2014), Chen *et al.* (2015a), we give the algorithm that can be implemented on SM. As shown in Figure 8, this algorithm utilizes an iterative method, which can make better use of sentence structure information.



```
Omni-segmentation(Seg_v, i){
(1) Get j from i+1 row;
(2) while(j>=i+1){
(3)     Seg_v.push_back(j)
(4)     if(n-1==j){
(5)         Collecting Seg_v as a SP;}
        else{
(6)         Omni-segmentation(Seg_v, j);}
(7)     j--;
}   }
```

Figure 8. Omni-Segmentation on SM

Suppose that one output of the kth iteration is $Seg\_v = (\cdots, i)$. In the $(k+1)$th iteration, the $(i+1)$th Row is processed. In Step (1), the largest *j* with $\mathbf{M}(i+1, j) = 1$ is obtained. In Step (2), add *j* into segmentation path Seg_v and judge whether or not j is the end of a sentence. If it equals $n-1$, then Seg_v is a segmentation path. If not, iterate this process. For all $\mathbf{M}(i+1, j) = 1$ in Row $i+1$, recycle from Step (3) Step (7).

This algorithm may generate tremendous segmentation paths. The combinational ambiguity can be filtered for simplicity.

4.3 Complexity

In order to identify CSA, both Omni-segmentation and BiMM methods try to segment a sentence for finding possible segmentation paths. The lexicon is required to be accessed frequently, and generated segmentation paths must be held for comparison. These processes involve massive string manipulations and string storages, leading to a higher computational and space complexity. In SM, after SM was built, string operations are mapped into set operations. There is no need for SM to implement string operations and access the lexicon. Moreover, SM can make better use of sentence structure information, decreasing the computational complexity. This section discusses the complexity of SM.

Let m to be the number of lexicon entry, n is the length of a given sentence, l is the length of the longest lexicon entry. In a given lexicon, l is a constant. The computational and space complexity are given as follows.

4.3.1 Computational Complexity

Searching for an element in a lexicon has computational complexity $O(\log m)$. Finding every word in a sentence need access lexicon $O(l \times n)$ times in the worst case. Therefore, the construction of SM has computational complexity $O(l \times n \times \log m)$. This is the same as FMM or BMM. Because BiMM implements both FMM and BMM, so BiMM has complexity $O(2 \times l \times n \times \log m + c)$. Where c denotes the complexity to compare the output of FMM and BMM.

In the worst scenario, SM has $O(l \times n)$ elements equal 1. In order to identify each overlapping ambiguity, for each $\mathbf{M}(i, j) = 1$, $O(l \times l)$ elements of M should be scanned. So identification of the OAS has complexity $O(l \times n \times \log m + l^3 \times n)$. Because l^2 and $\log m$ have the similar order, identification of OAS on SM has computational complexity close to BiMM.

4.3.2 Space Complexity

BiMM generates two FMM and BMM output. The space complexity for BiMM is constant. When Omni-segmentation is employed, segmentation path can grow tremendously, therefore, leading to a higher space complexity. In BiMM and Omni-segmentation, generated segmentation paths need to be held for OAS or CAS identification.

In the SM method, a $n \times n$ matrix is required. It seems that the space complexity for SM is $O(n^2)$. But in practical application, we deal with a sentence or sentence fragment. Storages of matrix can be used repeatedly. OAS or CAS can be identified directly without saving any segmentation path. Therefore, space complexity of SM also closes to BiMM.

5. Experiments

Before conducting experiments, we give an example to show the comparison between SM and BiMM. The sentence is given as "江泽民在北京人民大会堂会见参加全国法院工作会议和全国法院系统打击经济犯罪先进集体表彰大会代表时要求大家要充分认识打击经济犯罪工作的艰巨性和长期性". The outputs of FMM and BMM are listed as follows[10]:

S_F(0, 1, 2, 3, 5, 7, 9, 11, 12, 14, 16, 18, 20, 22, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 46, 48, 50, 51, 53, 55, 57, 59, 61, 62);

S_B(0, 1, 2, 3, 5, 7, 8, 10, 12, 14, 16, 18, 20, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 46, 48, 50, 51, 53, 55, 57, 59, 61, 62);

Making use of the BiMM method, 2 **MOAS** are detected: $MOAS[8,12]$, $MOAS[21,23]$. However, if SM method is employed, 5 **MOAS** and 10 **OAS** are found: $MOAS[8,12]$, $MOAS[15,18]$, $MOAS[21,23]$, $MOAS[24,27]$, $MOAS[38,41]$, $OAS[8,10]$, $OAS[9,11]$, $OAS[10,12]$, $OAS[15,17]$, $OAS[16,18]$, $OAS[21,23]$, $OAS[24,26]$, $OAS[25,27]$, $OAS[38,40]$, $OAS[39,41]$.

This example shows that BiMM is insufficient for OAS identification. To show more information, based on the Beijing University corpus (PKU corpus) of the Chinese word segmentation Bakeoff training data (Sproat *et al*., 2003), SM is compared with BiMM, DAG, and MNAG methods, as shown in Table 1.

Table 1. Comparison With Other Methods.

Model	MOAS		OAS		CAS	
	Type	Count	Type	Count	Type	Count
BiMM	8,409	19,090	×	×	×	×
DAG	7,369	12,337	18,888	51,895	38,200	515,151
MNAG	7,956	13,870	7,378	18,641	×	×
SM	19,269	52,072	26,580	101,514	39,310	555,574

Where, "×" means that this type of ambiguity cannot be identified by the corresponding method. It can be seen that SM shows better performance. Making use of the SM method, in the rest part of this section, several issues about CSA are discussed.

[10] The output may differ when different lexicon is adopted. In this place, we take the *Lexicon Common Words in Contemporary Chinese*.

In all experiments, we use the traditional $Precision$/$Recall$/$F\text{-}score$ (*P*/*R*/*F*) measurement to evaluate the performance. *Precision* is computed by $^{Correct\ Num}/_{Extract\ Num}$, and *Recall* is computed by $^{Correct\ Num}/_{Real\ Num}$. Where $Correct\ Num$ is the number of correctly recognized relation instances. $Extract\ Num$ is the number of instances that have been extracted. $Real\ Num$ refers to the number of annotated relation instances. F-score is computed by

$$\frac{2 \times (Precision \times Recall)}{Precision + Recall}$$

In the PKU corpus, the training data and testing data are provided. In the Penn Chinese Treebank corpus, the 5-fold cross validation is adopted for training and testing. We average the results of five runs as the final performance. To implement the maximum entropy classifiers, we used the toolkit provided by Zhang (2004). We also run a CRF model for comparison[11].

5.1 Characteristics of Chinese Segmentation Ambiguity

Characteristics of CSA have been investigated by other research. Sun *et al*. (1999) analyzed MOAS in a corpus containing 100 million characters. Li *et al.* (2003) studied 730,000 MOAS extracted from 20 years the *People's Daily* corpus. Li *et al*. (2006) collected 14,906 high frequent MOAS from the *People's Daily* corpus. Qiao *et al*. (2008) investigated MOAS in several corpora, which has more than 1 billion characters.

In general, these research is mainly focused on analysing MOAS for a given corpus. Rare research was conducted to study the characteristics of CSA in a given lexicon. This section is devoted to this. The *Lexicon of Common Words in Contemporary Chinese* is employed, which contains 56,008 words and is published by the *Ministry of Education of the People's Republic of China* in 2008. Table 2 shows detected segmentation ambiguity information.

Table 2. Ambiguity about Lexicon.

CAS Inside Word	39,944	OAS in Overlapping Words	1,847,814
OAS Inside Word	939	OAS in Adjacent Words	1,757,756
		Total OAS	1,847,814

CAS (OAS) Inside Word refers to overlapping (or combinational) ambiguity strings in a single word. *OAS in Overlapping Words* refers to overlapping ambiguity strings that are generated by overlapping two possible words. For example, “文科” (Liberal Arts) and “科学” (Science) can be overlapped to generate an OAS “文科学”. *OAS in Adjacent Words* denotes ambiguity strings generated by two adjacent words (no overlapping). For example, “点” (Point)

[11] *http://crfpp.googlecode.com/svn/trunk/doc/index.html.*

and “射门” (Shot) can be combined into an OAS “点射门”. It can be segmented as “点/射门” (Point/ Shot) or “点射/门” (Fixed Fire/ Door). *Total OAS Types* is produced by merging results in *OAS in Overlapping Words* and *OAS in Adjacent Words*. It can be seen as the overlapping ambiguity space.

As shown in Table 1, combinational ambiguities inside words are pervasive, even the Definition 4 is adopted. Except 2,927 words containing only single character, 75.25% words have combinational ambiguity. The *Total OAS Types* has the same number as *OAS in Overlapping Word*, so that *OAS in Adjacent Words* can be seen as a subset of *OAS in Overlapping Words*.

In the following, we investigate CSA in a large-scale corpus. The corpus contains 52,961 texts involving various literary genres. Because CSA cannot exist across punctuation. Therefore, instead of whole sentences, we take sentence fragments under consideration. After erasing duplicated sentence fragments, there are 0.2 billion sentence fragments remained. The information is shown in Table 3.

Table 3. Information of Corpus.

Corpus Size	8.26 Gigabyte	Texts	52,961
Total Characters	2,703,512,684	Total Words	1,902,306,846
Token	69,087	Sentence Fragments	264,748,094

Figure 9 shows the distribution of sentence fragment length in our corpus.

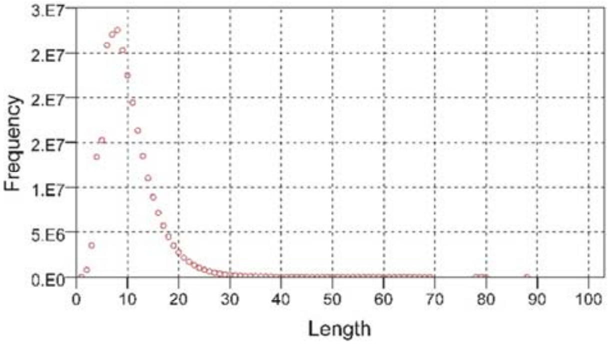


Figure 9. Distribution of Sentence Fragments Length

Almost 99% sentence fragments have length less than 26. Therefore, we set 128 as the size of SM. Longer sentence fragments are removed directly.

For each sentence fragment, Algorithm 1 and Algorithm 2 (See Section 4.1) are adopted to extract CAS and OAS. MOAS is obtained by merging OAS that are addable. These MOAS are referred to as *SM-MOAS*. In order to give a comparison, BiMM (See Section 4.2.3) is implemented to extract MOAS, referred as *BiMM-MOAS*. Table 4 gives information about CAS, OAS, *SM-MOAS* and *BiMM-MOAS*.

Table 4. Ambiguity about Corpus.

Ambiguity Type	CAS	OAS	SM-MOAS	BiMM-MOAS
Type	39,810	732,873	1,190,606	526,251
Count	579,238,862	40,921,520	32,641,105	23,424,525

Referring to Table 2, nearly all CAS types occurred in our corpus, but only 39.66% OAS types occurred. If the BiMM method is used, there are 91.52% sentence fragments having no overlapping ambiguity. If the SM method is employed, the rate reduces to 88.38%. It means that, by simple method, it is possible to get a massive sentence fragments without overlapping ambiguity for unsupervised methods (Li *et al*., 2003).

From Table 4, compared *SM-MOAS* and *BiMM-MOAS*, it can be seen that the number of *SM-MOAS* type is doubled. Therefore, only focusing on MOAS produced by BiMM is insufficient to study the CSA problem. In Figure 10, distributions of overlapping chain length about MOAS and OAS are compared.

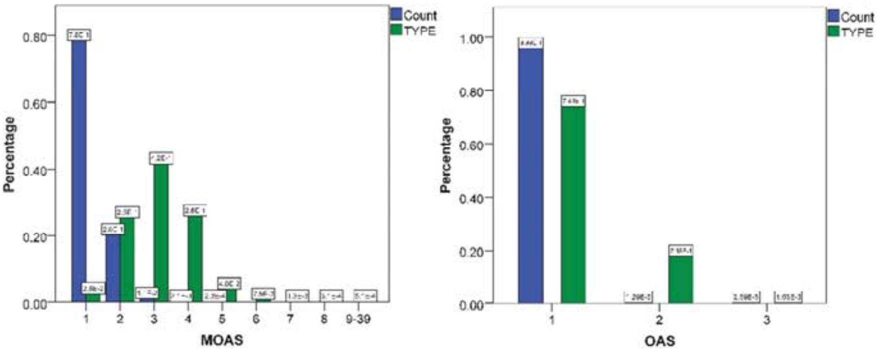
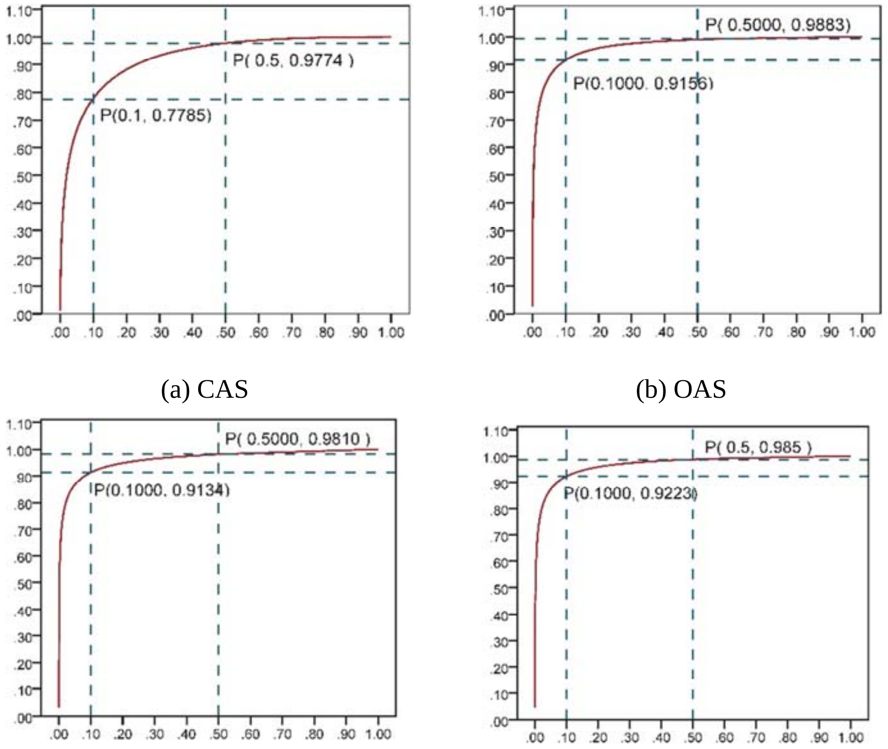


Figure 10. Distribution of Overlapping Chain Length

It can be seen that the distribution of overlapping chain length in MOAS is more complex, ranging from 1 to 39. However, it is simpler in OAS. There are 99.87% OAS has overlapping chain length equal to 1. This conclusion is useful for disambiguation. It can be modelled as a two-category classification problem.

Figure 11 shows the distributions of different CSA. X-axis represents the percentage of ambiguity string types in frequency-descending order. Y-axis is the percentage of occurrences. $P(x, y)$ represents x% of the highest frequency ambiguity string types covering y% occurrences. For each type of segmentation ambiguity, 10% high frequency ambiguity string types occupy 90% occurrences.



(a) CAS (b) OAS

(c) MOAS (d) BiMM MOAS

Figure 11. Distribution of Segmentation Ambiguity

5.2 Influence of Ambiguities on Word Segmentation

Based on FMM and BMM, Sun *et al.* (1995) analyzed the influence of CSA on Chinese word segmentation. Several conclusions were induced. These analyses mainly based on a corpus containing only 3,680 sentences. The influence of combinational ambiguity on Chinese word segmentation wasn’t studied. In this experiment, the Penn Chinese Treebank corpus[12] is used to analyze the influence of CSA on Chinese word segmentation. This corpus is manually segmented, consisting of 2,448 text files, 71,232 sentences, 1,196,329 words and 1,931,381 Hanzi (Chinese character). The segmentation ambiguity information is given in Table 5.

Table 5. Ambiguity about Penn Chinese TreeBank.

Ambiguity Type	CAS	OAS	SM-MOAS	BiMM-MOAS
Type	11,672	24,069	17,868	13,5591
Count	61,615	81,694	47,417	18,8275

[12] *http://www.cis.upenn.edu/~chinese/ctb.html*

Characteristics of CSA about the Penn Chinese Treebank are the same as our corpus discussed in Section 5.1.

Before given the experiment in detail, we first explain the terms used in this part. The meaning of ambiguity free has two levels. The first is that, for a given lexicon, a sentence has no segmentation ambiguity. The other is that a sentence may contain segmentation ambiguity that cannot be identified by an employed method. SM can identify every segmentation ambiguity. Therefore, *SM Free* means that a sentence contains no segmentation ambiguity at all. But *BiMM Free* is not. It only means that no segmentation ambiguity can be identified by the BiMM method.

Because sentence boundaries are manually labelled in the corpus, therefore, instead of segmentation fragments, the sentences are directly used as segmenting units, which contains 71,232 sentences. Among them, 56,618 sentences are *BiMM free*, and 43,444 sentences are *SM free*. Then, FMM or BMM is employed to segment collected sentences[13]. Performances are given in Table 6. Column 2 in Table 6 lists the number of selected sentences.

Table 6. Influence of OAS.

Method	Sentence	Precision	Recall	F1
FMM	71,232	95.08%	92.14%	93.59%
BMM	71,232	95.05%	92.09%	93.58%
BiMM Free	56,618	96.67%	93.61%	95.12%
SM Free	43,444	96.74%	93.68%	95.19%

Using FMM (or BMM) method, the performance is already upto 93.59% (93.58%) in F-score. In Row 3, if both FMM and BMM have the same output (*BiMM Free*), the precision is 96.67%. In Row 4, *SM free* means that there is no overlapping ambiguity at all, but segmentation precision is only 96.74%. Therefore, combinational ambiguity can cause segmentation errors about 3.3%.

5.3 Influence of Lexicon size on Chinese Word Segmentation

Making use of syntax and semantic knowledge, machine learning methods are successful to process the CSA problem. But these methods also have the disadvantage that annotated data is required, which is time-consuming and costly in human labor, and migrating between different applications is difficult. Lexicon-based method is easier to be implemented and has reasonable performance. Because only a lexicon is required for segmentation, the requirement for annotated data and the process for training are avoided. Therefore, the lexicon-based method

[13] The employed lexicon is directly extracted from the same corpus.

still be used in this field. In this section, the influence of lexicon size on Chinese word segmentation is studied. We use the PKU testing data. The FMM is used as the default method. This issue was discussed by other researchers (*e.g.* Sun *et al.*, (2001)), but no quantitative analysis is given. The result is shown in Table 7.

Table 7. Influence of Lexicon Size.

No.	Lexicon	Entries	Precision	Recall	F1
1	Testing Words	13,148	98.95%	98.62%	98.78%
2	Training Words	55,303	84.34%	90.77%	87.44%
3	(2) + CWCC	85,486	85.32%	90.23%	87.71%
4	(3) + Medium Lexicon	312,065	83.42%	83.16%	83.29%
5	(4) + Maximum Lexicon	554,331	81.23%	80.51%	80.87%
6	(5) + Testing Words	555,475	89.16%	83.49%	86.23%
7	(2) + Testing Words	58,166	97.26%	95.93%	96.59%

In Table 7, five lexicons are employed. *Testing Words* are words collected from the testing data. Words in *Training Words* are collected from training data. Performances generated by both are used as the topline and baseline in the Chinese word segmentation Bakeoff competition (Emerson, 2005). *CWCC* denotes the *Lexicon of Common Words* in *Contemporary Chinese*. *Medium Lexicon* is collected from the Internet, which contains 298,032 words. *Maximum Lexicon* is generated by merging *Medium Lexicon* and a *Great Dictionary of Chinese* with 542,240 lexicon.

In Chinese word segmentation, OOV (out-of-vocabulary) is considered as the main obstacle to segment a sentence (Sproat *et al.*, 2003; Huang *et al.*, 2007). Comparing Row 7 to Row 2 and Row 6 to Row 5, after testing words was added, the performances increases 9.15% and 5.36% respectively. When the lexicon size increased, the influence of OOV is slacked down. By comparing the Row 6 to Row 7, the lexicon used by Row 7 is a subset of Row 6, but Row 6 is lower than Row 7 about 10.36%. It is caused by overlapping and combinational ambiguities. Row 1 and Row 6 also have the same problem. Without segmentation disambiguation, increasing lexicon size can result in worse performance in lexicon-based methods. In order to see the influences in more details, Table 8 lists the number of errors caused in the segmentation.

Table 8. Information about Error.

No.	Lexicon	Total	By OAS	By CAS	By OOV
1	Testing Words	712	355	357	0
2	Training Words	7,386	752	1,298	5,336
3	(2) + CWCC	7,130	900	1,894	4,336
4	(3) + Medium Lexicon	9,695	1,595	5,433	2,667
5	(4) + Maximum Lexicon	10,954	2,363	5,982	2,609
6	(5) + Testing Words	8,110	2,088	6,022	0
7	(2) + Testing Words	2,045	697	1,348	0

In Table 8, the strategy to count the number of errors is explained as follows. If a segmentation path "A/ BC", is falsely segmented into "AB/ C" (A, B and C are characters). Then this failure is counted as an OAS error. If a segmentation path "A/ B" is falsely segmented into a word "AB" (combinational ambiguity), it is counted as a CAS error. An OOV error is caused by a word (*e.g*. "AB") falsely segmented into small pieces ("A/ B"). Figure 12 compares errors caused by OAS, CAS and OOV.

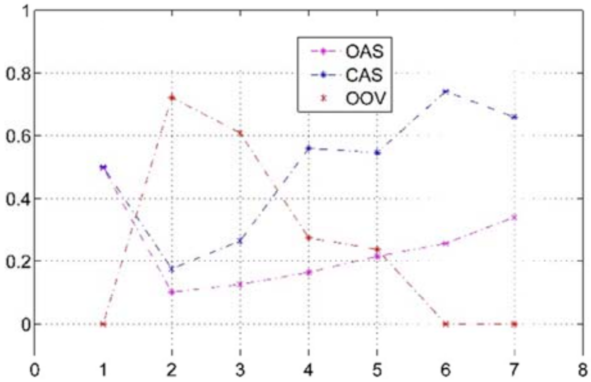


Figure 12. Influence of Lexicon Size on CSA and OOV

As shown in Figure 12, a larger dictionary decreases the OOV rate at the expense of increasing errors caused by OAS and CAS. When the size of the lexicon is large enough, without segmentation disambiguation, errors caused by CAS and OAS can exceed those caused by OOV. OAS is considered as a bottleneck of Chinese word segmentation. The result shows that, if the lexicon is large enough, the influence of CAS is the most critical. In practical applications, an encyclopedic dictionary with large number of lexicon entry is commonly adopted (Chien, 1997; Gao *et al*., 2002). The result indicates that the influence of CAS is important. For the lexicon-based method, increasing lexicon entry does not always guarantee better performance.

5.4 SM Segmentation

For traditional statistical-based methods, word segmentation is the way to find the segmentation path which has a maximized probability. Conditional Random Fields (CRF) received the state-of-the-art performances (Peng *et al*., 2004; Tseng *et al*., 2005). The proposed SM method is effective to identify lexical ambiguities, but weak for segmentation disambiguation. For segmenting a sentence based on SM, instead of finding a maximized segmentation path, the process can be divided into three steps: *OOV detection*, *OAS disambiguation* and *CAS disambiguation*. In the OOV detection step, new words are detected by an employed model[14], which reduces errors caused by the OOV problem. After sentences were segmented (*e.g*., by lexicon based method), the output can be further processed by OAS disambiguation and CAS disambiguation.

In this part, a preliminary experiment is given to demonstrate this process. The "closed test" is conducted based on the PKU corpus (Emerson, 2005). To make a comparison, a CRF model is implemented[15], which uses 3-Gram features and character features. The result is shown in Table 9, where Column *OAS* is the number of errors caused by OAS. Column *CAS* and Column *OOV* are the same.

Table 9. Performance On Segmentation.

Model	OAS	CAS	OOV	P	R	F1
FMM	752	1,298	5,336	84.34%	90.77%	87.44%
CRF	624	2,887	1,361	92.93%	91.32%	92.11%
SM+OOV	1,166	3,873	796	91.90%	88.84%	90.35%
SM+OOV+OAS	1,084	3,646	806	92.23%	89.37%	90.78%
SM+OOV+OAS+CAS	939	2,559	1,453	92.39%	91.14%	91.76%

In Table 9, the *SM+OOV* implements FMM, which uses words outputted by the CRF model (Row 2) and word extracted from training data. Comparing *SM+OOV* with FMM (Row 1), errors caused by OOV are reduced considerably. However, errors caused by OAS and CAS are increased. In the *SM+OOV+OAS*, another CRF model is trained only on OAS extracted from the training data, then implement the OAS disambiguation. In *SM+OOV+OAS+CAS*, a maximum entropy model is used to disambiguate CAS of *SM+OOV+OAS*. It also trained on

[14] In our experiment, we use CRF (trained on the training data) to segment the testing data, then collect generated new words.

[15] *http://crfpp.googlecode.com/svn/trunk/doc/index.html*

CAS extracted from training data. The result shows the performance is increased by OAS disambiguation and CAS disambiguation. Comparing with the CRF model, both show a similar performance.

Based on SM, the segmentation divides word segmentation into three steps. It provides an alternative way to process word segmentation. Making use of SM, each step can be optimized accordingly. However, the result also shows that the disambiguation of OAS and CAS are not independent. Decreasing one of them can influence the other. From Row 2 to Row 5, the CAS is also a challenging task for segmenting Chinese words.

6. Conclusions

In this paper, a SM method was provided, which represents lexicon information as a matrix. Under the framework of set theory, formal definitions about *segmentation path, combinational ambiguity, overlapping ambiguity*, etc. are given. By mapping string operations into set operation, SM is effective in CSA identification and also available for Chinese word segmentation. In our experiments, several issues about CSA were explored. For researchers who are interested in our work, the source code of our SM is available at *https://github.com/YPench/SMatrix/*.

7. Acknowledges

The research was supported in part by the National Science Foundation of China under Grant Nos. 91118005, 91218301, 91418205.

Reference

Chang, C. & Wei, J. (2008). Identification and disposal of ambiguity based on Omni-Segmentation arithmetic. *Computer Engineering and Applications*, 15.

Chen, K. & Bai, M. (1998).Unknown word detection for Chinese by a corpus-based learning method. *International Journal of Computational Linguistics and Chinese Language Processing*, *3*(1), 27-44.

Chen, X., Li, L., & Liang, X. (2012). Eliminate Semantic Network Word Segmentation Ambiguity Method Research. *Microelectronics & Computer*, *3*, 045.

Chen, Y., Zheng, Q., & Zhang, W. (2014). Omni-word Feature and Soft Constraint for Chinese Relation Extraction. *The Proceedings of ACL'14*, 572-581.

Chen, Y., Zheng, Q., & Chen, P. (2015a). Feature assembly method for extracting relations in Chinese. *Artificial Intelligence*, *228*, 179-194.

Chen, Y., Zheng, Q., & Chen, P. (2015b). A Boundary Assembling Method for Chinese Entity-Mention Recognition. *Intelligent Systems, IEEE, 30*(6), 50-58.

Chien, L. (1997). PAT-tree-based keyword extraction for Chinese information retrieval. *ACM SIGIR Forum, 31*(SI), 50-58.

Emerson, T. (2005). The second international chinese word segmentation bakeoff. *The Proceedings of SIGHAN '05*, 133.

Gao, J., Goodman, J., Li, M., & Lee, K. (2002).Toward a unified approach to statistical language modeling for Chinese. *ACM Transactions on Asian Language Information Processing*, *1*(1), 3-33.

Gao, D., & Guo, J. (2009). Dealing with Chinese Overlapping Ambiguity Based on Type Functional Application. *The Proceedings of AICI '09*, *3*, 67-71.

Gao, Y., He, J., & Li, J. (2011). Research on Chinese phonetic string segmentation of sentential input. *The Proceedings of CECNet '11*, 4334-4337.

Gao, J., Li, M., Wu, A., & Huang, C. (2005). Chinese word segmentation and named entity recognition: A pragmatic approach. *Computational Linguistics*, *31*(4), 531-574.

Hatori, J., Matsuzaki, T., Miyao, Y., & Tsujii, J. (2012). Incremental joint approach to word segmentation, POS tagging, and dependency parsing in Chinese. *The Proceedings of ACL '12*, *1*, 1045-1053.

Huang, C., & Zhao, H. (2007). Chinese Word Segmentation: A Decade Review. *Journal of Chinese Information Processing*, *21*(3), 8-19.

Jiang, W., Mi, H., & Liu, Q. (2008). Word lattice reranking for Chinese word segmentation and part-of-speech tagging. *The Proceedings of COLING '08*, *1*, 385-392.

Li, B. (2011). *Research on Chinese Word Segmentation and proposals for improvement*. (Master thesis). Roskilde University, Denmark.

Li, M., Gao, J., Huang, C., & Li, J. (2003). Unsupervised training for overlapping ambiguity resolution in Chinese word segmentation. *The Proceedings of SIGHAN '03*, *17*, 1-7.

Li, Z., & Sun, M. (2009). Punctuation as implicit annotations for Chinese word segmentation. *Computational Linguistics*, *35*(4), 505-512.

Li, Z., Zhang, M., Che, W., Liu, T., *et al*. (2011). Joint models for Chinese POS tagging and dependency parsing. *The Proceedings of EMNLP '11*, 1180-1191.

Liang, N. (1984). Written Chinese word segmentation system-CDWS. *Journal of Beijing Institute of Aeronautics and Astronautics*, *4*, 97-104.

Luo, X., Sun, M., & Tsou, B. (2002).Covering ambiguity resolution in Chinese word segmentation based on contextual information. *The Proceedings of COLING '02*, *1*, 1-7.

Peng, F., Feng, F., & McCallum, A. (2004). Chinese segmentation and new word detection using conditional random fields. *The Proceedings of COLING '04*, 562.

Qiao, W., Sun, M., & Menzel, W. (2008). Statistical properties of overlapping ambiguities in Chinese word segmentation and a strategy for their disambiguation. *Text, Speech and Dialogue*, 177-186.

Sproat, R., & Emerson, T. (2003). The first international Chinese word segmentation bakeoff. *The Proceedings of SIGHAN '03*, *17*, 133-143.

Sun, W. (2011). A stacked sub-word model for joint Chinese word segmentation and part-of-speech tagging. *The Proceedings of ACL '11, 1,* 1385-1394.

Sun, M., & Benjamin K (1995). Ambiguity Resolution in Chinese Word Segmentation. *The Proceedings of PACLIC '95*.

Sun, T., Liu, Y., Yang, L., *et al*. (2009). An ambiguity discovery algorithm on Chinese word segmentation based dictionary. *The Proceedings of WMWA '09,* 39-42.

Sun, M., & Tsou, B. (2001). A review and evaluation on automatic segmentation of Chinese. *Contemporary Linguistics, 3*(1), 22-32.

Sun, M., & Zou, J. (2001). A critical appraisal of the research on Chinese word segmentation. *Contemporary Linguistics, 1,* 002.

Sun, M., Zuo, Z., & Tsou, B. (1999). The role of high frequent maximal crossing ambiguities in Chinese word segmentation. *Journal of Chinese information processing, 13*(1), 27-37.

Teahan, W., Wen, Y., McNab, R., & Witten, I. (2000). A compression-based algorithm for Chinese word segmentation. *Computational Linguistics, 26*(3), 375-393.

Tseng, H., Chang, P., Andrew, G., *et al*. (2005). A conditional random field word segmenter for sighan bakeoff 2005. *The Proceedings of SIGHAN '05*, 171.

Wand, S., & Wang, B. (2007). A Chinese Overlapping Ambiguity Resolution Method Based on Coupling Degree of Double Characters. *Journal of Chinese Information Processing, 5,* 004.

Wang, X., & Du, L. (2004). A Method of Sentence Segmentation That Check All Overlapping Ambiguity. *Acta Electronica Sinica, 32*(1), 50-54.

Wang, L., Song, S., Feng, J., & Chen, P. (2009). Chinese Segmentation System Combining Omni-Segmentation with Statistic. *Microelectronics & Computer, 5*, 019.

Wang, K., Zong, C., & Su, K. (2012). Integrating generative and discriminative character-based models for Chinese word segmentation. *ACM Transactions on Asian Language Information Processing, 11*(2), 7.

Wang, Z., Zong, C., & Xue, N. (2013). A Lattice-based Framework for Joint Chinese Word Segmentation, POS Tagging and Parsing. *The Proceedings of ACL '13,* 2013.

Wu, A., & Jiang, Z. (1998). Word segmentation in sentence analysis. *The Proceedings of ICCIP '98*, 169-180.

Wu, X., Zhang, M., & Lin, X. (2010). Parsing-based Chinese word segmentation integrating morphological and syntactic information. *The Proceedings of NLP-KE '11,* 114-121.

Xue, N. (2003). Chinese word segmentation as character tagging. *Computational Linguistics and Chinese Language Processing, 8*(1), 29-48.

Xue, N., & Yang, Y. (2011). Chinese sentence segmentation as comma classification. *The Proceedings of ACL '11, 2*, 631-635.

Yao, H., Wang, Y., & Huang, J. (2012). An algorithm of solving Chinese segmentation overlapping ambiguous. *The Proceedings of CSAE '12, 2*, 464-467.

Zeng, D., Wei, D., Chau, M., & Wang, F. (2011). Domain-specific Chinese word segmentation using suffix tree and mutual information. *Information Systems Frontiers*, *13*(1), 115-125.

Zeng, X., Wong, D., Chao, L., & Trancoso, I. (2013). Graph-based Semi-Supervised Model for Joint Chinese Word Segmentation and Part-of-Speech Tagging. *The Proceeding of ACL '13*, 770-779.

Zhang, L. (2004). *Maximum entropy modeling toolkit for Python and C++*. Natural Language Processing Lab, Northeastern University, China.

Zhang, P., & Li, C. (2006). A new Context-Sensitive ambiguous phrase segmentation Algorithm. *Computer Systems & Applications*, *5*, 013.

Zhang, L., Li, L., He, Z., *et al.* (2013). Improving Chinese Word Segmentation on Micro-blog Using Rich Punctuations. *The Proceeding of ACL '13*.

Zhang, H., & Liu, Q. (2002). Model of Chinese Words Rough Segmentation Based on N-Shortest-Paths Method. *Journal of Chinese information processing*, *5*, 000.

Zhang, H., Yu, H., Xiong, D., & Liu, Q. (2003). HHMM-based Chinese lexical analyzer ICTCLAS. *The Proceedings of SIGHAN '03*, *17*, 184-187.

Zhao, H., Huang, C., Li, M., & Lu, B. (2010). A unified character-based tagging framework for Chinese word segmentation. *ACM Transactions on Asian Language Information Processing*, *9*(2), 5.