

Chinese Word Segmentation by Mining Maximized Substrings

Mo Shen, Daisuke Kawahara, and Sadao Kurohashi

Graduate School of Informatics, Kyoto University

Yoshida-honmachi, Sakyo-ku,

Kyoto, 606-8501, Japan

shen@nlp.ist.i.kyoto-u.ac.jp {dk,kuro}@i.kyoto-u.ac.jp

Abstract

A major problem in the field of Chinese word segmentation is the identification of out-of-vocabulary words. We propose a simple yet effective approach for extracting maximized substrings, which provide good estimations of unknown word boundaries. We also develop a new semi-supervised segmentation technique that incorporates retrieved substrings using discriminative learning. The effectiveness of this novel approach is demonstrated through experiments using both in-domain and out-of-domain data.

1. Introduction

Chinese sentences are written without explicit word boundaries, which makes Chinese word segmentation (CWS) an initial and important step in Chinese language processing. Recent advances in machine learning techniques have boosted the performance of CWS systems. On the other hand, a major difficulty in CWS is the problem of identifying out-of-vocabulary (OOV) words, as the Chinese language is continually and rapidly evolving, particularly with the rapid growth of the internet.

A recent line of research to overcome this difficulty is through exploiting characteristics of frequent substrings in unlabeled data. Statistical criteria for measuring the likelihood of a substring being a word have been proposed in previous studies of unsupervised segmentation, such as *accessor variety* (Feng et al., 2004) and *branching entropy* (Jin and Tanaka-Ishii, 2006). This kind of criteria has been applied to enhance the performance of supervised segmentation systems (Zhao and Kit, 2007; Zhao and Kit, 2008;

Substring	Freq
一致	3
界限数的期望值	2
一致认定界限	2
的期望值	3
认定界限数的	2
值	4

Table 1. A particular type of substrings with multiple occurrences in the Chinese sentence: “使一致认定界限数的期望值近似于一致正确界限数的期望值，求得一致认定界限的期望值/认定界限数的值。”

Sun and Xu, 2011) by identifying unknown word boundaries.

In this paper, instead of investigating statistical characteristics of batched substrings, we propose a novel method that extracts substrings as reliable word boundary estimations. The technique uses large-scale unlabeled data, and processes it on the fly.

To illustrate the idea, we first consider the following example taken from a scientific text:

“使一致认定界限数的期望值近似于一致正确界限数的期望值，求得一致认定界限的期望值/认定界限数的值。”

Without any knowledge of the Chinese language one may still notice that some substrings like “一致” and “的期望值”, occur multiple times in the sentence and are likely to be valid words or chains of words. Consider a particular type of frequent substring that cannot be simultaneously extended by its surrounding characters while still being equal (Table 1). We can observe that the boundaries of such substrings can be used as perfect word delimiters. We can segment the sentence by simply treating the boundaries of each occurrence of a substring in Table 1 as word

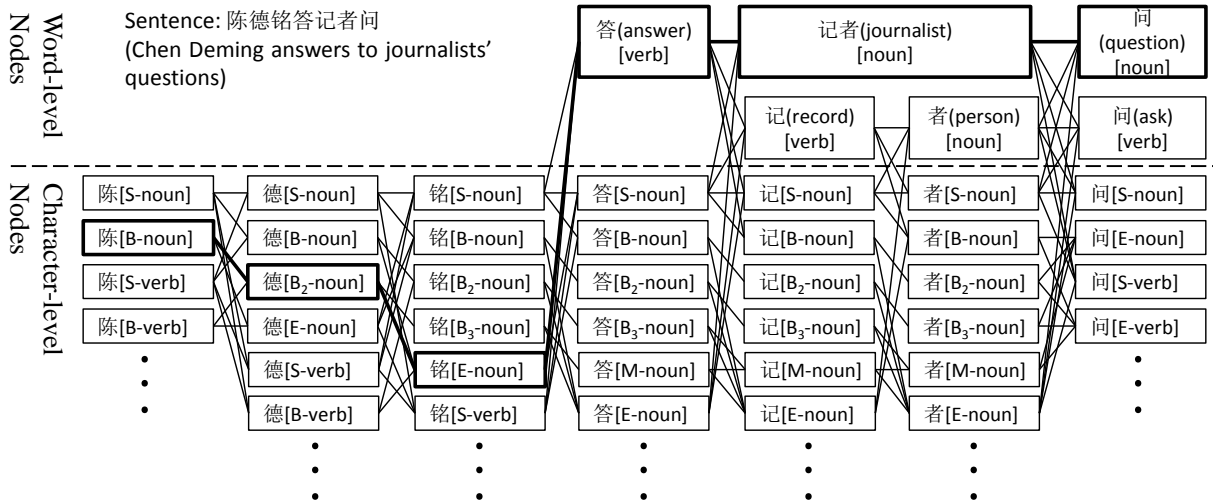


Figure 1. A Word-character hybrid lattice of a Chinese sentence. Correct path is represented by bold lines.

Word Length	1	2	3	4	5	6	7 or more
Tags	<i>S</i>	<i>BE</i>	<i>BB₂E</i>	<i>BB₂B₃E</i>	<i>BB₂B₃ME</i>	<i>BB₂B₃MME</i>	<i>BB₂B₃M...ME</i>

Table 2. Word representation with a 6-tag tagset: *S, B, B₂, B₃, M, E*

delimiters:

“使|一致|认定|界限|数|的|期望|值|近似于|一致|正确|界限数|的|期望|值|，求得|一致|认定|界限|的|期望|值|/|认定|界限|数的|值|。”

Compared with the gold-standard segmentation, this partial segmentation has a precision of 100% and a recall of 73.3% with regard to boundary estimation. This is high when we consider that the method does not use a trained segmenter or annotated data. While we have obtained this result on a selected instance, it still suggests that unlabeled data has the potential to enhance the performance of supervised segmentation systems by tracking consistency among substrings.

Substrings, such as those listed in Table 1, are retrievable from unlabeled data and can be incorporated with a supervised CWS system to compensate for out-of-vocabulary (OOV) words. In this case the unlabeled data can be either test data only (leading to a purely supervised system), or a large-scale external corpus (leading to a semi-supervised system). We will formally define this particular type of substring, referred to as a “maximized substring”, in a later section.

The remainder of this paper is organized as follows. Section 2 describes our baseline segmentation system, defines maximized substrings, and proposes an efficient algorithm for retrieving these substrings from unlabeled data. Section 3

introduces the maximized substring features. Section 4 presents the experimental results. Section 5 discusses related work. The final section summarizes our conclusions.

2. Approach

2.1 Baseline Segmentation System

We have used a word-character hybrid model as our baseline Chinese word segmentation system (Nakagawa and Uchimoto, 2007; Kruengkrai et al., 2009). As shown in Figure 1, this hybrid model constructs a lattice that consists of word-level and character-level nodes from a given input sentence. Word-level nodes correspond to words found in the system’s lexicon, which has been compiled from training data. Character-level nodes have special tags called position-of-character (POC) that indicate the word-internal position (Asahara, 2003; Nakagawa, 2004). We have adopted the 6-tag tagset, which (Zhao et al., 2006) reported to be optimal. This tagset is illustrated in Table 2.

Previous studies have shown that jointly processing word segmentation and part-of-speech tagging is preferable to separate processing, which can propagate errors (Nakagawa and Uchimoto, 2007; Kruengkrai et al., 2009). If the training data was annotated by part-of-speech tags, we have combined them with both word-level and character-level nodes.

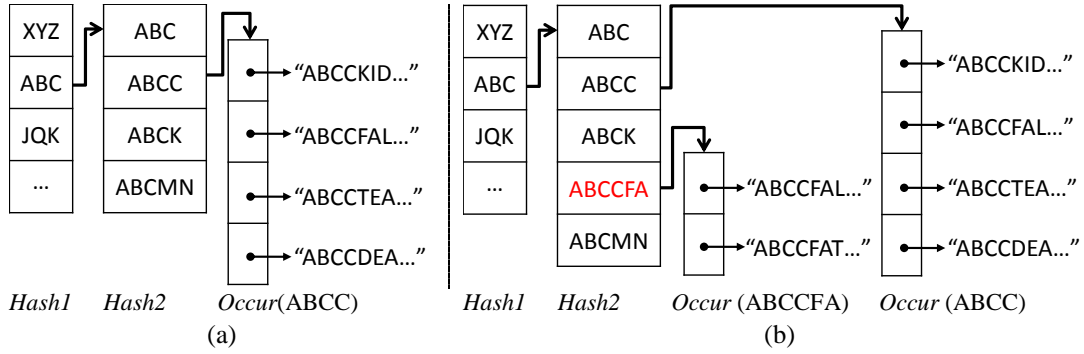


Figure 2. Data structure for maximized substring mining. *Hash1* is the first-level hash with fixed-length prefix keys. *Hash2* is a hash associating to a corresponding key in *Hash1* that stores the list of *maximized substrings* sharing the same fixed-length prefix. *Occur*(\cdot) is the occurrence list associating to a particular *maximized substrings* with references to all its occurrences in the original positions in the document. (a) shows a certain state of the data structure, and (b) the state after a *maximized substring* “ABCCFA” is inserted with the context being “ABCCFAT...” in the document.

2.2 Maximized Substring: the Definition

Frequent substrings in unlabeled data can be used as clues for identifying word boundaries, as we have illustrated in Section 1. Nevertheless, some substrings, although frequent, are not useful to the system. In the example in Section 1, the substring “致认定界” occurs the same amount of times as the substring “一致认定界限”. However, only the latter is a valid identifier for word delimiters: they are non-overlapping, meaning that it is impossible to simultaneously extend all occurrences by surrounding characters. We use the term *maximized substring* to describe these substrings.

Formally, we define *maximized substring* as follows:

Definition 1 (Maximised substring). Given a document D that is a collection of sentences, denote a length n substring which starts with character c_t by $s_t = [c_t c_{t+1} \dots c_{t+n-1}]$. s_t is called a *maximized substring* if:

1. It has a set of distinct occurrences, M , with at least two elements¹:

$$M = \{s_{t_1}, s_{t_2}, \dots, s_{t_m}\}, \quad m > 1, \quad t_1 \neq t_2 \neq \dots \neq t_m \text{ s.t. } s_{t_1} = s_{t_2} = \dots = s_{t_m};$$
 and
2. $c_{t_i-1} \neq c_{t_j-1}$ and $c_{t_i+n} \neq c_{t_j+n} \quad \forall i, j = 1, 2, \dots, m, i \neq j$.

¹ It should be noted that, in order to retrieve a substring, the size of M is not necessarily identical to its total count in the document.

The substrings listed in Table 1 are therefore maximized substrings, given that D is the example sentence. Note that these are *not all* maximized substrings extractable from the example sentence, but are the result of the retrieval algorithm that we will describe in the next section.

2.3 Maximized Substring Retrieval: Algorithm and Data Structure

The problem of mining frequent substrings in a document has been extensively researched. Existing algorithms generally either use a suffix tree structure (Nelson, 1996) or suffix arrays (Fischer et al., 2005), and make use of the apriori property (Agrawal and Srikant, 1994). The apriori property states that a string of length $k+1$ is frequent only if its substring of length k is frequent. The apriori property can significantly reduce the size of enumerable substring candidates. However, as we are only interested in maximized substrings, suffix tree-based algorithms are inefficient in both time and space. We therefore propose a novel algorithm and a compact data structure for fast maximized substring mining.

The data structure is illustrated in Figure 2. It supports fast prefix searching for storing and retrieving maximized substrings, with each entry associated to a list of occurrences that refer to the original positions in the document. Fast prefix matching is a particular advantage of a trie, which is a type of prefix tree. Our structure is different as we use a two-level hash structure for space efficiency and ease of manipulation. This is important, especially during experiments on large-scale unlabeled data.

The first-level hash stores prefixes of a fixed-length, m , of retrieved substrings. This part of the data structure functions as a filter to screen

out substrings that are shorter than m characters, as they should not be considered as candidates. This is motivated by our observation that single characters, and sometimes even double-character substrings, are not reliable enough to predict word delimiters. Note that m is data dependent, for example, the optimal value of m is 3 characters on the dataset Chinese Treebank (CTB).

Each key of the first-level hash is associated with a second-level hash that stores the retrieved maximized substrings that share a common prefix.

The third-level structure is a linked list of occurrences of a particular maximized substring. This list stores references to the original position of each occurrence of the substring, with the surrounding context being visible so that new (longer) maximized substrings can be found by extension.

We sketch the process of maximized substring retrieval in Pseudocode 1. From the beginning of the document D , we scan each position and register maximized substrings into the data structure H . If an incoming substring already exists in H , we look up its occurrence list to check if its succeeding characters can extend the substring. As the current occurrence list is a set of maximized substrings, there will be only two possible outcomes. Either exactly one element in the occurrence list is found to have a longer common prefix with the incoming substring, in which case we create a new occurrence list consisting of the two lengthened substrings. Alternatively, the prefix remains the same and we add the incoming substring to the occurrence list.

We can easily demonstrate that all substrings retrieved by this algorithm are maximized substrings. However, the algorithm does not generally guarantee to retrieve all maximized substrings from unlabeled data. This is a necessary compromise if we wish to keep the efficiency of one-time scanning. In addition, we have observed in preliminary experiments that retrieving all maximized substrings is not only unnecessary, but can introduce harmful noise. In the next section, we will discuss our solution to this problem.

2.4 Short-Term Store

Maximized substrings can provide good estimations of word boundaries, but random noise can be introduced during the retrieval process in Pseudocode 1.

To address this problem, we take advantage of a linguistic phenomenon. It has been observed that a word occurring in the recent past has a

Pseudocode 1: Maximized substring retrieval

```

1  procedure RetrieveMaxSub( $m, D$ )
2     $i \leftarrow 0, H \leftarrow \emptyset$ 
3     $p \leftarrow [c_0 c_1 \dots c_{m-1}]$ 
4     $\triangleleft$  the reference of a length  $m$  substring at
5      the beginning of document  $D$ 
6    until  $i$  reaches the end of document  $D$ 
7       $s \leftarrow$  longest element in  $H$  extendable
8        from  $p$ 
9      if  $|s| = 0$   $\triangleleft$  empty string
10        $occurList \leftarrow \{p\}$ 
11        $\triangleleft$  make the occurrence list of  $p$ 
12        $H.Add(\langle p, occurList \rangle)$ 
13        $\triangleleft$  associate  $p$  with its occurrence list
14         and add to data structure
15        $i \leftarrow i + 1$ 
16        $p \leftarrow [c_i \dots c_{i+m-1}]$ 
17     else
18        $p^* \leftarrow [c_i \dots c_{i+|s|-1}]$ 
19        $(i, p) \leftarrow \text{Maximize}(H, m, i, s, p^*)$ 
20     return  $H$ 
21
22  procedure Maximize( $H, m, i, s, p^*$ )
23    for each  $e$  in  $s.occurList$ 
24       $\langle s_{new}, e_{new}, p_{new}^* \rangle \leftarrow \text{Extend}(e, p^*)$ 
25       $\triangleleft$  find the longest common substring
26         $s_{new}$  between  $e$  and  $p^*$  by simultane-
27        ously extending them with succeeding
28        characters
29      if  $|s_{new}| > |s|$ 
30         $occurList_{new} \leftarrow \{e_{new}, p_{new}^*\}$ 
31         $H.Add(\langle s_{new}, occurList_{new} \rangle)$ 
32         $i \leftarrow i + |s_{new}|$ 
33         $p \leftarrow [c_i \dots c_{i+m-1}]$ 
34      return  $(i, p)$ 
35  end
36   $s.occurList.Add(p^*)$ 
37   $i \leftarrow i + |s|$ 
38   $p \leftarrow [c_i \dots c_{i+m-1}]$ 
39  return  $(i, p)$ 

```

much higher probability to occur again soon, when compared with its overall frequency (Kuhn and Mori, 1990). It follows that, for speech recognition, we can then use a window of recent history to adjust the static overall language mode.

This observation is applicable to the task of maximized substring retrieval in the following way. Suppose a substring is registered into the data structure. If the substring is in fact a word, it is much more likely to reoccur in the next 50 to 100 sentences than in the remainder of the corpus (especially when it is a technical term or a named entity). Otherwise the substring should have a more unified probability of reoccurrence across the entire corpus.

This motivated us to introduce a functionality into the process of maximized substring retrieval, called “short-term store” (STS). The STS is an analogy to the cache component in speech recognition as well as the human phonological working memory in language acquisition. It restricts the length of the visible context when retrieving the next candidate of a registered substring, making it proportional to the current number of occurrences of the substring. For a registered substring, the retrieval algorithm scans a certain number of sentences after the latest occurrence of the substring, where the number of sentences $D(s)$ is determined as follows:

$$D(s) = \begin{cases} \lambda \cdot \text{count}(s), & \text{if } \text{count}(s) < \theta, \\ \infty, & \text{otherwise,} \end{cases}$$

where $\text{count}(s)$ is the current number of occurrences of s in the data structure. The parameter λ contributes a fixed-length distance to the visible context. The parameter θ works as a threshold of reliability. If we have observed s at least θ times in a short period, we can regard s as a word, or a sequence of words, with a high level of confidence. Thus, $D(s) = \infty$ implies that s is no longer subject to periodical decaying and will stay in the data structure statically.

During the scanning of the $D(s)$ sentences, if a new occurrence of s is found, it is added into the data structure and $D(s)$ is recalculated immediately, starting a new scanning period. If no new occurrences are found, we remove the earliest occurrence of s from the data structure and then re-calculate $D(s)$. Note that we have described the short-term store functionality as if each substring in the data structure is scanned separately. In practice, however, only a small change to Pseudocode 1 is required so that STS is used, making one-time scanning of the unlabeled data sufficient.

Introducing STS into the retrieval process results in a substantial improvement to the quality of retrieved substrings. It is also important that STS greatly improves the processing efficiency for large scale unlabeled data by keeping the size of the data structure relatively small. This is because a substring entry will decay from the data structure if it has not been refreshed in a short period.

3. Features

3.1 Baseline Features

For baseline features, we apply the feature tem-

plates described in (Kruengkrai et al., 2009). For further details, please see the original paper. Note that if the part-of-speech tags are not available, we omit those templates involving POS tags.

3.2 Maximized Substring Features

We have incorporated the list of retrieved maximized substrings into the baseline system by using a technique which discriminatively learns their features. For every word-level and character-level node in the lattice, the method checks the maximized substring list for entries that satisfy the following two conditions:

1. The node matches the maximized substring at the beginning, the end, or both boundaries.
2. The length of the node is shorter than or equal to that of the entry.

For example, consider the lattice in Figure 1 with a maximized substring “陈德铭”. All of the character-level nodes of “陈” and “铭” are encoded with maximized substring features. A segmenter will only obtain information on those possible word boundaries that are identified by maximized substrings. The maximized substrings are not directly treated as single words, because a maximized substring can sometimes be a compound word or phrase.

For each match with a maximized substring entry, the technique encodes the following features.

Basic: A binary feature that indicates whether the match is at the beginning or end of the maximized substring. It is encoded both individually and as a combination with each other feature types.

Lexicon: There is a particular kind of noise in the retrieved list of maximized substrings, namely, those like the substring “中美经”, which has resulted from the two phrases “中美经济” (China and U.S. economy) and “中美经贸” (China and U.S. economic and trade). This happens when the boundary of a maximized substring is a shared boundary character of multiple other words. In this example, the last character “经” of the maximized substring is the character at the beginning of “经济” (economy) and “经贸” (economic and trade). This kind of noise can be identified by checking the context of maximized substrings in system’s lexicon.

Our technique checks the context of the maximized substring in the input sentence and compares it with the system’s lexicon. If any item in the lexicon is found that forms a positional relation with the maximized substring entry (as listed

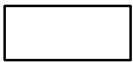

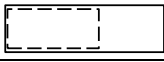
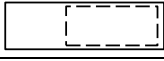

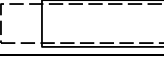

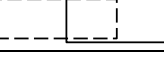


Sentence: $s = \dots c_{-1}c_0c_1 \dots c_{n-1}c_nc_{n+1} \dots$		Representa- tion
Maximized substring $m = c_0c_1c_2 \dots c_n$		
Lexicon entry $l = c_ic_{i+1}c_{i+2} \dots c_j$		
ID	Positional Relation	
L1	$0 = i < j < n$	
L2	$0 < i < j = n$	
L3	$0 = i < n < j$	
L4	$i < 0 < j = n$	
L5	$0 < i < n < j$	
L6	$i < 0 < j < n$	
L7	$i = n + 1$	
L8	$j = -1$	

Table 3. Lexicon features. Each one represents a positional relation between a maximized substring and a contextual substring which exists in system’s lexicon.

ID	At Beginning	ID	At Ending
B1	<L1,L6>	E1	<L2,L5>
B2	<L6,L8>	E2	<L5,L7>
B3	<L1,L8>	E3	<L2,L7>

Table 4. Lexicon Composition features. Each one represents a combination of two Lexicon features that fire simultaneously.

in Table 3) then the corresponding features are encoded.

Lexicon Composition: When a maximized substring is a match to more than one item in the lexicon, a combination of multiple lexicon features is more informative than individual features. We encode the combinations of lexicon features listed as in Table 4.

Frequency: We sort the list of maximized substrings by their frequencies. If a maximized substring is among the 10% most frequent it is classed as “highly frequent”, if it is among the top 30% it is “normal”, and all other cases are “infrequent”.

4. Evaluation

4.1 Setting

To evaluate our approach, we have conducted word segmentation experiments on two datasets. The first is Chinese Treebank 7 (CTB7), which is a widely used version of the Penn Chinese Treebank dataset for the evaluations of word segmentation techniques. We have adopted the same setting of data division as (Wang et al., 2011): the training set, dev set and test set. For CTB7, these sets have 31,131, 10,136 and 10,180 sentences respectively. The second dataset is the second international Chinese word segmentation bakeoff (SIGHAN Bakeoff-2005) (Emerson, 2005), which has four independent subsets: the Academia Sinica Corpus (AS), the Microsoft Research Corpus (MSR), the Hong Kong City University Corpus (CityU) and the Peking University Corpus (PKU). Since POS tags are not available in this dataset, we have omitted all templates that include them. The models and parameters applied on all test sets are those that result in the best performance on the CTB7 dev set.

We have used two different types of unlabeled data. One is the test set itself, which means the system is purely supervised. Another is a large-scale dataset, which is the Chinese Gigaword Second Edition (LDC2007T03). This dataset is a collection of news articles from 1991 to 2004 published by Central News Agency (Taiwan), Xinhua News Agency and Lianhe Zaobao Newspaper. It includes a total amount of over 1.2 billion characters in both simplified Chinese and traditional Chinese.

We have trained all models using the averaged perceptron algorithm (Collins, 2002), which we selected because of its efficiency and stability. To learn the characteristics of unknown words, we built the system’s lexicon using only the words in the training data with a frequency higher than a threshold, h . This threshold was tuned using the development data. In order to use the maximized substring features, we have used training data as unlabeled data for supervised models, and used both the training data and Chinese Gigaword for semi-supervised models.

We have applied the same parameters for all models, which are tuned on the CTB7 dev set: $m = 3$, $h = 2$, $\lambda = 100$, and $\theta = 3$.

We have used precision, recall and the F-score to measure the performance of segmentation systems. Precision, p , is defined as the percentage of

System	P	R	F
Baseline	95.17	95.35	95.26
MaxSub-Test	95.33	95.47	95.40
MaxSub-U	95.65	95.81	95.73

Table 5. Evaluation on CTB7 for the baseline approach and our approach with small and large-scale in-domain unlabeled data respectively.

words that are segmented correctly, and recall, r , is the percentage of words in the gold standard data that are recognized in the output. The balanced F-score is defined as $F = 2pr/(p + r)$.

4.2 Experimental Results on In-domain Data

We have compared the performance between the baseline system and our approach. The results are shown in Table 5. Each row in this table shows the performance of the corresponding system. “Baseline” refers to our baseline hybrid word segmentation and POS-tagging system. “MaxSub-Test” refers to the method that just uses the test set as unlabeled data. “MaxSub-U” refers to the method that uses the large-scale unlabeled data. We have focused on the segmentation performance of our systems.

The results show that, using the test data as an additional source of information, “MaxSub-Test” outperforms the baseline method by 0.14 points in F-score. This indicates that our method of using maximized substrings can enhance the segmentation performance even with a purely supervised approach. The improvement increases to 0.47 points in F-score for “MaxSub-U”, which demonstrates the effectiveness of using large-scale unlabeled data.

We have compared our approach with previous work in Table 6. Two methods from (Kruengkrai et al., 2009a; 2009b) are referred to as “Kruengkrai 09a” and “Kruengkrai 09b”, and are taken directly from the report of (Wang et al., 2011). “Wang 11” refers to the semi-supervised system in (Wang et al., 2011). We have observed that our system “MaxSub-U” achieves the best segmentation among these systems. Also, although the performance of our baseline is lower than the systems “Kruengkrai 09a” and “Kruengkrai 09b” because of differences in implementation, the system “MaxSub-Test” (which has used no external resource) has achieved a comparable result.

The results for the SIGHAN Bakeoff-2005 dataset are shown in Table 7. The first three rows (“Tseng 05”, “Asahara 05” and “Chen 05”) show the results of systems that have reached the highest score on at least one corpus (Tseng et al.,

System	F
Baseline	95.26
MaxSub-Test	95.40
MaxSub-U ⁺	95.73
Kruengkrai 09a	95.40
Kruengkrai 09b	95.46
Wang 11 ⁺	95.65

Table 6. F-measure on CTB7 test set compared with previous work. “⁺”: semi-supervised systems.

System	AS	CityU	MSR	PKU
Tseng 05	94.7	94.3	96.4	95.0
Asahara 05	95.2	94.1	95.8	94.1
Chen 05	94.5	94.0	96.0	95.0
Best closed	95.2	94.3	96.4	95.0
Zhang 07	95.1	95.1	97.2	95.1
Zhao 07	95.5	95.6	97.5	95.4
Baseline	95.07	94.53	96.25	95.13
MaxSub-S	95.17	94.61	96.42	95.31
MaxSub-L ⁺	95.34	94.79	96.64	95.55

Table 7. F-measure on SIGHAN Bakeoff-2005 test set compared with previous work. “⁺”: semi-supervised systems.

2005; Asahara et al., 2005; Chen et al., 2005). “Best closed” summarizes the best official results on all four corpora. “Zhao 07” and “Zhang 06” represent the supervised segmentation systems in (Zhao and Kit, 2007; Zhang et al., 2006). “Baseline”, “Maxsub-Test” and “MaxSub-U” refer to the same systems as in Table 5. For the unlabeled data, we have used the test sets of corresponding corpora for “MaxSub-Test”, and the Chinese Gigaword for “MaxSub-U”. Other parameters were left unchanged. The results do not indicate that our approach performs better than other systems. However, this is largely because of our baseline not being optimized for these corpora. Nevertheless, when compared with the baseline, our approach has yielded consistent improvements across the four corpora, and on the PKU corpus we have performed better than previous work.

4.3 Impacts of Semi-supervised Features and Short-term Store

In Table 8, we have shown the effects of the different maximized substring feature types proposed in this paper. We activated different combinations of feature types in turn and trained separate models. We also investigated the impact of the short-term store by training models without this feature. The rows of this table represent models and corresponding F-measure, trained

System	F
Baseline	95.26
+Basic&Freq	95.50
+All	95.60
+All+STS	95.73

Table 8. Influence of activated feature types and short-term store on CTB7 test data.

System	P	R	F
Baseline	91.88	92.02	91.95
MaxSub-Test	92.43	92.53	92.48

Table 9. Results on out-of-domain data.

and tested on CTB7 with different configurations. The row “Baseline” is baseline system as in Table 5. “+Basic&Freq” represents the system “MaxSub-U” with only basic and frequency features activated, and STS turned off. The row “+All” represents a system activating all maximized substring features but still without STS. The last row “+All+STS” is identical to the system “Maxsub-U”. It is clear that lexicon-based features are effective in discriminating unreliable maximized substring from reliable ones, and the short-term store improves the segmentation performance by filtering out noises during the retrieval of maximized substrings. The combination of these two techniques yields an improvement of 0.23 point in F-measure, and thus are essential when using maximized substrings.

4.4 Experimental Results on Out-of-domain Data

To demonstrate the effectiveness of our method on out-of-domain text, we have conducted an experiment on a test set that was drawn from a corpus of scientific articles. This test set contains 510 sentences that have been manually segmented by a native Chinese speaker. We used the test set as the unlabeled data.

As the results show (Table 9), the system “MaxSub-Test” exceeded the baseline method by 0.53 in F-score, which is a significant improvement. Considering that the amount of unlabeled data is relatively small, it is likely that acquiring large-scale unlabeled data in the same domain will further benefit the accuracy.

5. Related Work

The authors of (Feng et al., 2004) proposed accessor variety (AV), a criterion measuring the likelihood of a substring being a word by count-

ing distinct surrounding characters. In (Jin and Tanaka-Ishii, 2006) the researchers proposed branching entropy, a similar criterion based on the assumption that the uncertainty of surrounding characters of a substring peaks at the word boundaries. The authors of (Zhao and Kit, 2007) incorporated accessor variety and another type of criteria, called co-occurrence sub-sequence, with a supervised segmentation system and conducted comprehensive experiments to investigate their impacts. Although the idea behind co-occurrence sub-sequence is similar with maximized substrings, there are several restrictions: it requires post-processing to remove overlapping instances; sub-sequences are retrievable only from different sentences; and the retrieval is performed only on training and testing data. In (Sun and Xu, 2011), the authors proposed a semi-supervised segmentation system enhanced with multiple statistical criteria. Large-scale unlabeled data were used in their experiments.

Li and Sun presented a model to learn features of word delimiters from punctuation marks in (Li and Sun, 2009). Wang et al. proposed a semi-supervised word segmentation method that took advantages from auto-analyzed data (Wang et al., 2011).

Nakagawa showed the advantage of the hybrid model combining both character-level information and word-level information in Chinese and Japanese word segmentation (Nakagawa, 2004). In (Nakagawa and Uchimoto, 2007) and (Kruengkrai et al., 2009a; 2009b) the researchers presented word-character hybrid models for joint word segmentation and POS tagging, and achieved the state-of-the-art accuracy on Chinese and Japanese datasets.

6. Conclusion

We propose a simple yet effective approach for extracting maximized substrings from unlabeled data. These are a particular type of substrings that provide good estimations of unknown word boundaries. The retrieved maximized substrings are incorporated with a supervised segmentation system through discriminative learning. We have demonstrated the effectiveness of our approach through experiments in both in-domain and out-of-domain data and have achieved significant improvements over the baseline systems across all datasets².

² $p < 0.05$ in McNemar’s test.

References

- Rakesh Agrawal and Ramakrishnan Srikant. 1994. Fast Algorithms for Mining Association Rules. In Proceedings of 1994 Int. Conf. Very Large Data Bases, pages 487–499.
- Masayuki Asahara. 2003. Corpus-based Japanese Morphological Analysis. Nara Institute of Science and Technology, Doctor's Thesis.
- Masayuki Asahara, Kenta Fukuoka, Ai Azuma, Chooi-Ling Goh, Yotaro Watanabe, Yuji Matsumoto, and Takashi Tsuzuki. 2005. Combination of Machine Learning Methods for Optimum Chinese Word Segmentation. In Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing, pages 134–137.
- Michael Collins. 2002. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In Proceedings of EMNLP 2002, pages 1–8.
- Thomas Emerson. 2005. The Second International Chinese Word Segmentation Bakeoff. In Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing, pages 123–133.
- Aitao Chen, Yiping Zhou, Anne Zhang, and Gordon Sun. 2005. Unigram language model for Chinese word segmentation. In Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing, pages 138–141.
- Haodi Feng, Kang Chen, Xiaotie Deng, and Weimin Zheng. 2004. Accessor Variety Criteria for Chinese Word Extraction. *Computational Linguistics*, 30(1), pages 75–93.
- Johannes Fischer, Volker Heun, and Stefan Kramer. 2005. Fast Frequent String Mining Using Suffix Arrays. In Proceedings of ICDM 2005, IEEE Computer Society, pages 609–612.
- Zhihui Jin and Kumiko Tanaka-Ishii. 2006. Unsupervised Segmentation of Chinese Text by Use of Branching Entropy. In Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 428–435.
- Canasai Kruengkrai, Kiyotaka Uchimoto, Jun'ichi Kazama, YiuWang, Kentaro Torisawa, and Hitoshi Isahara. 2009. An Error-Driven Word-Character Hybrid Model for Joint Chinese Word Segmentation and POS Tagging. In Proceedings of ACL/IJCNLP 2009, pages 513–521.
- Canasai Kruengkrai Kiyotaka Uchimoto, Jun'ichi Kazama, Yiu Wang, Kentaro Torisawa, and Hitoshi Isahara. 2009. Joint Chinese Word Segmentation and POS Tagging Using an Error-Driven Word-Character Hybrid Model. *IEICE transactions on information and systems*, 92(12), pages 2298–2305.
- Roland Kuhn and Renato De Mori. 1990. A Cache-based Natural Language Model for Speech Recognition. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 12(6), pages 570–583.
- Zhongguo Li and Maosong Sun. 2009. Punctuation as Implicit Annotations for Chinese Word Segmentation. *Computational Linguistics*, 35(4), pages 505–512.
- Mark Nelson. 1996. Fast String Searching with Suffix Trees. *Dr.Dobb's Journal*.
- Tetsuji Nakagawa. 2004. Chinese and Japanese Word Segmentation Using Word-level and Character-level Information. In Proceedings of COLING 2004, pages 466–472.
- Tetsuji Nakagawa and Kiyotaka Uchimoto. 2007. Hybrid Approach to Word Segmentation and Pos Tagging. In Proceedings of ACL 2007 Demo and Poster Sessions, pages 217–220.
- Yiou Wang, Jun'ichi Kazama, Yoshimasa Tsuruoka, Wenliang Chen, Yujie Zhang, and Kentaro Torisawa. 2011. Improving Chinese Word Segmentation and POS Tagging with Semi-supervised Methods Using Large Auto-Analyzed Data. In Proceedings of IJCNLP 2011.
- Weiwei Sun and Jia Xu. 2011. Enhancing Chinese Word Segmentation Using Unlabeled Data. In Proceedings of EMNLP 2011, pages 970–979.
- Huihsin Tseng, Pichuan Chang, Galen Andrew, Daniel Jurafsky, and Christopher Manning. 2005. A Conditional Random Field Word Segmenter for SIGHAN Bakeoff 2005. In Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing, pages 168–171.
- Hai Zhao, Chang-Ning Huang, Mu Li, and Bao-Liang Lu. 2006. Effective Tag Set Selection in Chinese Word Segmentation via Conditional Random Field Modeling. In Proceedings of PACLIC 20, pages 87–94.
- Hai Zhao and Chunyu Kit. 2007. Incorporating Global Information into Supervised Learning for Chinese Word Segmentation. In Proceedings of PACLING 2007, pages 66–74.
- Hai Zhao and Chunyu Kit. 2008. Exploiting Unlabeled Text with Different Unsupervised Segmentation Criteria for Chinese Word Segmentation. *Research in Computing Science*, Vol. 33, pages 93–104.
- Ruiqiang Zhang, Genichiro Kikui, and Eiichiro Sumita. 2006. Subword-based Tagging for Confidence Dependent Chinese Word Segmentation. In COLING/ACL 2006, pages 961–968.