Chinese Word Segmentation with Minimal Linguistic Knowledge: An Improved Conditional Random Fields Coupled with Character Clustering and Automatically Discovered Template Matching

Richard Tzong-Han Tsai, Hong-Jie Dai, Hsieh-Chuan Hung, Cheng-Lung Sung,
Min-Yuh Day, and Wen-Lian Hsu, Fellow, IEEE
Academia Sinica, Taiwan, R.O.C.
{*thtsai*, *hongjie*, *yabt*, *clsung*, *myday, hsu*}*@iis.sinica.edu.tw*

Abstract

This paper addresses three major problems of closed task Chinese word segmentation (CWS): word overlap, tagging sentences interspersed with non-Chinese words, and long named entity (NE) identification. For the first, we use additional bigram features to approximate trigram and tetragram features. For the second, we first apply K-means clustering to identify non-Chinese characters. Then, we employ a two-tagger architecture: one for Chinese text and the other for non-Chinese text. Finally, we post-process our CWS output using automatically generated templates. Our results show that additional bigrams can effectively identify more unknown words. Secondly, using our two-tagger method, segmentation performance on sentences containing non-Chinese words is significantly improved when non-Chinese characters are sparse in the training corpus. Lastly, identification of long NEs and long words is also enhanced by template-based post-processing. Using corpora in closed task of SIGHAN CWS, our best system achieves F-scores of 0.956, 0.947, and 0.965 on the AS, HK, and MSR corpora respectively, compared to the best context scores of 0.952, 0.943, and 0.964 in SIGHAN Bakeoff 2005. In AS, this performance is comparable to the best result (F=0.956) in the open task.

1. Introduction

The amount of electronic text in Chinese has increased dramatically. The ability to search, index, and process such text is therefore in high demand. The first obstacle we face in processing Chinese text is Chinese word segmentation (CWS) problem. In western languages, words are delimited by separators (i.e. spaces), while in Chinese they are not. There has been significant research into finding word boundaries in unsegmented sequences and several popular annotated corpora also exist. However, it is very difficult to define a consistent and unified standard, and each corpus uses different word segmentation standards.

An important problem is word overlap. In many situations, several possibilities may seem correct. For example, the trigram "中国立" in the sentence "中国立国五千年, " (China has had a continuous civilization for five thousand years) can be interpreted in two ways. If the current character is "国", both "中国" (China) and "国立" (national) are valid words. In such limited contexts, a CWS system that can weigh all *n*-grams in the window and assign tags according to corpus statistics would be useful. A popular CWS approach is machine learning (ML) such as Conditional Random Fields (CRF) [1] .

The second problem is identifying and segmenting non-Chinese word sequences in Chinese documents, especially in a closed task (to be described later). A good CWS system needs to handle Chinese texts peppered with non-Chinese words (e.g., Arabic numbers or English words). Since their morphologies are quite different from Chinese morphology, our approach must depend on how many non-Chinese words appear, whether they connect with each other, and whether they are interleaved with Chinese words. If we can distinguish these non-Chinese characters automatically and apply different strategies, the performance can be improved.

The third problem in CWS, that of correctly identifying longer named entities (NEs), also stems from the limited context window of ML-based CWS. Most ML-based CWS systems use a five-character context window to determine the current character's tag. However, these systems often handle long words poorly, such as dates, locations, and institute names. For example, "中国科技信息所 (Institute of Science & Technical Information of China) 在 (under) 国家科委(National Nature Science of China) 支持 (support) " may be incorrectly segmented as "中国科技/信息/所在/国家科委 /支持".

The recent SIGHAN CWS bakeoff contest [2] has facilitated system development and comparison. It provides

0-7803-9788-6/06/$20.00 ©2006 IEEE.

four datasets with significantly different segmentation guidelines and consistent train-test splits. The contest divides its tasks as closed and open. In closed tests, participants were only allowed to use information found in the training data. Absolutely no other data or information could be used beyond that in the training document. This included knowledge of character sets, punctuation characters, etc. These seemingly artificial restrictions were formulated to study exactly how far one can get without supplemental information. In the open tests participants could use any external data in addition to the training corpus to train their system. We follow the rule of the closed task because we want construct a CWS system with minimal linguistic knowledge and compare with other state-of-the-art systems with the same goal.

2. Chinese Word Segmentation System

2.1. Conditional Random Fields

CRFs are undirected graphical models trained to maximize a conditional probability [3]. A linear-chain CRF with parameters $\Lambda=\{\lambda_1, \lambda_2, \ldots\}$ defines a conditional probability for a state sequence $\mathbf{y} = y_1 \ldots y_T$ given an input sequence $\mathbf{x} = x_1 \ldots x_T$ to be

$$P_\Lambda(\mathbf{y} \mid \mathbf{x}) = \frac{1}{Z_\mathbf{x}} \exp\left(\sum_{t=1}^{T}\sum_{k} \lambda_k f_k(y_{t-1}, y_t, \mathbf{x}, t)\right) \quad (1),$$

where Z_x is the normalization that makes the probability of all state sequences sum to one; $f_k(y_{t-1}, y_t, \mathrm{x}, t)$ is often a binary-valued feature function and λ_k is its weight. The feature functions can measure any aspect of a state transition, $y_{t-1} \rightarrow y_t$, and the entire observation sequence, $\mathbf{x}$, centered at the current time step, t. For example, one feature function might have value 1 when y_{t-1} is the state B, y_t is the state I, and x_t is the character "国". Large positive values for λ_k indicate a preference for such an event; large negative values make the event unlikely.

The most probable label sequence for $\mathbf{x}$,

$$\mathbf{y}^* = \arg\max_{y} P_\Lambda(\mathbf{y} \mid \mathbf{x}),$$

can be determined using the Viterbi algorithm [4].

2.2. N-gram Features

Character n-gram features are commonly used and have proven their effectiveness in ML-based CWS. As mentioned above, the ability to weigh all n-grams in the window and assign tags according to corpus statistics is very important. In our CRF-based CWS, the n-gram features are represented using first and second order state transition feature functions [9]. Specifically, we use four types of unigram feature functions, designated as C_0 (current character), C_1 (next character), C_{-1} (previous character), C_{-2} (the one two characters back). Furthermore, six types of bigram features are used, $C_{-2}C_{-1}$, $C_{-1}C_0$, C_0C_1, $C_{-3}C_{-1}$, $C_{-2}C_0$, $C_{-1}C_1$. Our approach is the first to use $C_{-3}C_{-1}$ and $C_{-2}C_0$ in a CWS system. Consider 埋头苦"干" (concentrate on "working") and 埋头苦"思" (concentrate on "thinking"). The two Chinese idioms shares the same prefix "埋头苦" (concentrate), therefore, if one of them is in the training set, and the other is in the test set, then the bigram feature, where "C_{-3}='埋' and C_{-1}= '苦'", will be an effective feature for tagging C_0.

2.3. Character Clustering

In many cases, Chinese sentences may be interspersed with non-Chinese words. Since it is not allowed, we have no way of knowing exactly how many languages are in a given text in the closed task. Our solution is to apply a clustering algorithm to find homogeneous characters belonging to the same character clusters. The first difficulty is how to represent each character in the vector space. One general rule we can follow is that a language's characters will tend to appear together in tokens. In addition, character clusters exhibit certain distinct properties. The first property is that the order of some pairs of characters can be exchanged, either before or after the other. This is referred to as *exchangeability*. The second property is that some characters can appear in any position of a word, such as lowercase characters; while others cannot, such as uppercase characters. This is referred to as *locational independence*. According to the general rule, we can calculate pairing frequency of characters in tokens by checking all tokens in the corpus. Assuming the alphabet is Σ, we first need to represent each character as a $|\Sigma|$ dimensional vector. For each character c_i, we use v_j to represent its j-dimension value, which is calculated as follows:

$$v_j = \alpha + (1-\alpha)[\min(f_{ij}, f_{ji})]^r \quad (2),$$

where f_{ij} stands for the frequency with which c_i and c_j appear in the same word when c_i's position precedes that of c_j. We take the minimum value of f_{ij} and f_{ji} because even when c_i and c_j have a high co-occurrence frequency, if either f_{ij} or f_{ji} is low, then one order does not exist, and v_j's value will be low. We use two parameters to normalize v_j to the range between 0 and 1. Parameter α enlarges the gap between non-zero and zero frequencies, and γ weakens the influence of very high frequencies.

Next, we apply the K-means algorithm to generate candidate sets that are composed of K clusters [5]. Different K's, α's, and γ's are used to generate possible character cluster sets. Our K-means uses cosine distance.

After obtaining the K clusters, we need to select the N_1 best character clusters from among them. First, we remove the one-character clusters. Then, assuming the angle between the cluster centroid vector and $(1, 1, \cdots, 1)$ is θ, the cluster that has the largest cosine θ will be re-

moved. This is because characters whose co-occurrence frequencies are nearly all zero will be transformed into vectors very close to (α, α, $\cdots$,α). Therefore, their centroids will also be very close to (α,α, $\cdots$,α), leading to unreasonable results.

Then, for each character c in cluster M, we calculate the inverse relative distance (IRDist) of c using (3):

$$\mathrm{IRDist}(c) = \log\left(\sum_i \cos(c, m_i) / \cos(c, m)\right) \quad (3),$$

where m_i and m are the centroids of M_i and M.

Then, we calculate the average inverse distance for each M. The N_1 best are selected from the original K.

The above K-means clustering and character cluster selection is executed iteratively for each cluster set generated from K-means with different K's, α's, and γ's.

After selecting the N_1 best clusters for each cluster set, we add them all to a pool and rank them according to their inner ratio, which is calculated as follows:

$$\mathrm{inner}(M) = \sum_{c_i, c_j \in M} \mathrm{co-occur}(c_i, c_j) / \sum_{c_i, c_j} \mathrm{co-occur}(c_i, c_j) \quad (4),$$

where co-occur(c_i, c_j) stands for the frequency with which character c_i and c_j co-occur in the same word.

Algorithm 1 Balanced Cluster Selection

Input: A set of character clusters $P=\{M_1, \ldots, M_K\}$
Number of selections N_2,
Output: A set of clusters $Q=\{M'_1, \ldots, M'_{N_2}\}$.

```
1: C={}
2: sort the clusters in P by their inner ratio;
3: while |C|<=N2 do
4:    pick the cluster M that has highest inner ratio;
5:    for each character c in M do
6:       if frequency of c in C∪M is over threshold τ
7:             P←P−M;
8:             continue;
9 :      else
10:            C←C∪M;
11:            P←P−M;
12:      end;
13:  end;
14: end
```

To ensure that we select a balanced mix of clusters, for each character in the incoming cluster M, we use Algorithm 1 to check if the frequency of each character in $C \cup M$ is greater than a threshold τ.

The above algorithms give us best N_2 clusters in terms of exchangeability.

Then we execute the above procedures again to select the N_2 best clusters for both locational independence and exchangeability. The only difference now is that for each character c_i, we use v_j to stand for its j dimension value. We calculate v_j as follows:

$$v'_j = \alpha + (1-\alpha)[\min(\overline{f_{ij}}, f'_{ij}, \overline{f_{ji}}, f'_{ji})]^r \quad (5)$$

where $\overline{f_{ij}}$ stands for the frequency with which c_i and c_j appear in the same word when c_i is the first character. f'_{ij} stands for the frequency with which c_i and c_j co-occur in the same word when c_i precedes c_j but not in the first position. We choose the minimum value from $\overline{f_{ij}}$, f'_{ij}, $\overline{f_{ji}}$, and f'_{ji} since if c_i and c_j can both appear in the first position of a word and their order is exchangeable, the four frequency values will all be large enough, including the minimum value.

Our next goal is to create the best hybrid of the above two cluster sets. The set selected for exchangeability is referred to as set EX, and that selected for both exchangeability and locational independence as set EL. We create a development set and use the best first strategy to build the optimal cluster set from EX∪EL. The EX and EL for HK corpus are shown in Table 1. Section 2.4 details how we exploit character cluster information in CWS.

Table 1 Clustering results of HK corpus

	Cluster	Inner	(K, α, γ)
EX	,.0123456789	0.94	(10, 0.6, 0.16)
	-/ABCDEFGHIKLMNOPRSTUVWabcdefghiklmnoprstuvwxy	0.93	(10, 0.7, 0.16)
EL	－／ABCDEFGHIKLMNOPRSTUVWabcdefghiklmnoprstvwxy	0.84	(10, 0.5, 0.25)
	０１２３４５６７８９	0.76	(10, 0.5, 0.26)

2.4. Handling Non-Chinese Words

In Chinese texts interspersed with non-Chinese words and phrases, non-Chinese characters suffer from serious data sparseness problem since their frequencies are much lower than Chinese characters. As to the bigrams containing at least one non-Chinese character (referred as non-Chinese bigrams), the problem becomes more serious. Take the phrase "約莫 20 歲" (about 20 years old) for example. The character "2" is usually predicted as *I*, (i.e., "約莫" is connected with "2") and result in a wrong segmentation, because the frequency of "2" in the *I* class is much higher than that of "2" in the *B* class even though the feature $C_{-2}C_{-1}$="約莫" has high weight for assigning "2" to the *B*.

Traditional CWS approaches use one general tagger (referred as G tagger). In our system, we use two. Similar to the traditional approach, we still have a general tagger. In addition to that, we design a specialized tagger to deal with non-Chinese words. The composite tagger (general plus specialized tagger) is referred as GS tagger.

Section 2.3 has described the details of how to generate character clusters. Here, all characters in the selected clusters are referred as non-Chinese characters. In the development stage, the best-first feature selector will determine which clusters are used. Then, we convert each sen-

tence in the training and test data into normalized sentence. Each non-Chinese character c is replaced by a cluster representative symbol σ_M, where c is in cluster M. We refer the string composed of all σ_M as F. If $|F|$ is greater than W, it will be shortened to length W. Then, the normalized sentence is placed in one file, and the non-Chinese character sequence is placed in another. Then, we use the normalized training and test file for the general tagger, and the non-Chinese sequence training and test file for the specialized tagger. Finally, the results of these two taggers are combined.

The advantage is that the data sparseness of non-Chinese bigrams is relieved. Consider the previous example. σ stands for the numeral cluster. Since "約莫 8 年" is not in the training data, $C_{-1}C_0$= "莫 8" is still an unknown bigram using G tagger. But using GS tagger, "約莫 20 歲 " and "約莫 8 年" will be converted as "約莫 σσ 歲" and "約莫 σ 年", respectively. Therefore, the bigram feature $C_{-1}C_0$="莫 σ" is no longer unknown. And since σ in "莫 σ" is tagged as B, (i.e., "莫" and "σ" are separated), "莫" and "σ" are separated in "約莫 σσ 歲".

2.5. Generating Long-word Templates

2.5.1. Global Alignment and Template Generation. Since long words have certain morphological similarities, we use global alignment to generate long-word templates. Take the two NEs "中国三峡总公司" and "中国核工业总公司" for example. They have common affixes "中国" and "总公司." Therefore, we can apply global alignment to generate the template "中国∗总公司".

2.5.2. Template Generation. We first extract all possible word candidates from the training set. Given a minimum word length L, we extract all words whose length is greater than or equal to L. Then we align all word pairs. For each pair, if more than 50% of the chars are identical, a template will be generated to match both of them.

2.5.3. Template Filtering. We have two criteria to filter the extracted templates. First, we test each template t's accuracy in the development set, which is as follows:

$$A(t) = \frac{\#\text{ of matched strings with no separators}}{\#\text{ of all matched strings}}$$

In our system, templates that have accuracy lower than the threshold τ_1 are directly discarded. For the remaining templates, we apply two different strategies. Most templates with accuracy under τ_2 are ineffective. To refine such templates we employ the character class information generated by character clustering to impose class limitation to certain template slots. This regulates potential input and improves precision. Consider a template with one or more wildcard slots. If any string matched with these wildcard slots contain characters in different clusters, this template is also discarded.

2.5.4. Template-Based Post-Processing (TBPP). The generated templates are then used to match our CWS output and check if the matched tokens can be combined into complete words. If a template's accuracy is greater than τ_2, then all separators in the matched strings will be eliminated. Otherwise, for a template t with accuracy between τ_1 and τ_2, we will eliminate all separators in its matched string if there is no substring matched with the t's wildcard slots contains characters in different clusters. Resulting words shorter than three characters are discarded because CRF performs well with such words.

3. Experiment

3.1. Dataset

We use the three corpora in SIGHAN Bakeoff 2005: a Simplified Chinese corpus provided by Microsoft Research Beijing and two Traditional Chinese corpora provided by Academia Sinica and the City University of Hong Kong. Details on are provided in Table 3.

Table 3. Corpora information

Corpus	Training Size		Test Size	
	Types	Words	Types	Words
Academia Sinica (AS)	141 K	5.45 M	19 K	122 K
City University (HK)	69 K	1.46 M	9 K	41 K
Microsoft Research (MSR)	88 K	2.37 M	13 K	107 K

3.2. Results of Baseline Tagger

The n-grams used by [1] is used as our baseline n-grams, as shown in Table 4. Detailed figures are shown in Table 5. R_{OOV} stands for the recall rate of the out-of-vocabulary words. R_{IV} stands for the recall rate of the in-vocabulary words, and NC stands for n-changes.

Table 4. Baseline features

Uni-gram	Bigram
C_{-2}, C_{-1}, C_0, C_1	$C_{-2}C_{-1}$, $C_{-1}C_0$, C_0C_1, $C_{-1}C_1$, C_0C_2

Table 5. Detailed baseline performance

Corpus	R	P	F	R_{OOV}	R_{IV}	NC
AS	0.949	0.957	0.953	0.706	0.969	7750
HK	0.948	0.944	0.946	0.726	0.963	2819
MSR	0.964	0.960	0.962	0.687	0.968	5468

3.3. Results of Best G Tagger

The best n-gram features are listed in Table 6.

Table 6. Best *n*-gram features

Uni-gram	Bigram
C_{-2}, C_{-1}, C_0, C_1	$C_{-2}C_{-1}$, $C_{-1}C_0$, C_0C_1, $C_{-3}C_{-1}$, $C_{-2}C_0$, $C_{-1}C_1$

3.4. Results of GS Tagger

Table 7. Performance on sentences with non-Chinese words

Conf	R	P	F	R_{OOV}	R_{IV}	NC	NCR
AS_G	0.917	0.920	0.918	0.708	0.943	928	7.54%
AS_{GS}	0.932	0.918	0.925	0.744	0.955	858	
HK_G	0.945	0.951	0.948	0.729	0.961	1040	2.88%
HK_{GS}	0.948	0.952	0.950	0.744	0.966	1010	
MSR_G	0.949	0.956	0.952	0.757	0.959	1964	5.66%
MSR_{GS}	0.951	0.959	0.955	0.792	0.960	1853	

Table 8. Performance on sentences without non-Chinese words

Conf	R	P	F	R_{OOV}	R_{IV}	NC	NCR
AS_G	0.961	0.952	0.957	0.725	0.971	6635	1.16%
AS_{GS}	0.961	0.953	0.957	0.732	0.971	6558	
HK_G	0.942	0.946	0.944	0.728	0.959	1768	-0.91%
HK_{GS}	0.942	0.946	0.944	0.735	0.960	1784	
MSR_G	0.965	0.971	0.968	0.682	0.971	3267	-1.13%
MSR_{GS}	0.965	0.971	0.968	0.675	0.970	3304	

Table 9. Overall performance

Conf	R	P	F	R_{OOV}	R_{IV}	NC	NCR
AS_G	0.950	0.958	0.954	0.722	0.969	7563	1.94%
AS_{GS}	0.951	0.959	0.955	0.732	0.970	7416	
HK_G	0.948	0.944	0.946	0.737	0.962	2809	0.53%
HK_{GS}	0.948	0.944	0.946	0.739	0.963	2794	
MSR_G	0.967	0.960	0.963	0.721	0.967	5231	1.43%
MSR_{GS}	0.967	0.961	0.964	0.737	0.968	5156	

Table 7 and 8 compare GS and G tagger's performance on sentences with and without non-Chinese characters respectively. ″NCR″ stands for the NC reduction rate of using GS tagger. Table 9 compares them on the whole corpora. We can see that the GS tagger not only improves non-Chinese-sentence and overall performance, but also improves performance on Chinese-only sentences in AS. Figure 1 shows the NC reduction attained by using the GS tagger. NC reduction in AS is more dramatic due to ability of the GS tagger to handle sparse distribution of non-Chinese characters. In AS's training set, there are only 6.8% sentences containing non-Chinese words, which is much less than HK (38.8%) and MSR (28.9%)

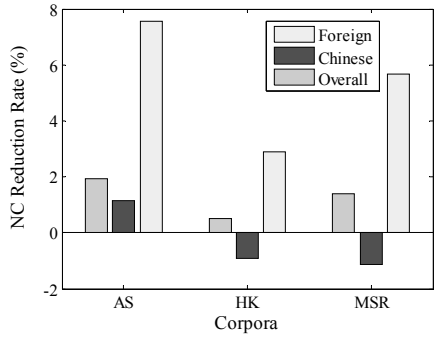

Figure 1. Comparison of NC Reduction

3.5. Results of TBPP

Next, we evaluate the performance of TBPP on the GS tagger's output, as shown in Table 11.

Table 11. Detailed TBPP performance

	R	P	F	R_{OOV}	R_{IV}	NC
AS	0.952	0.960	0.956	0.737	0.970	7270
HK	0.949	0.945	0.947	0.740	0.963	2768
MSR	0.969	0.962	0.965	0.751	0.968	4957

Figure 2 illustrates that our system outperforms the best SIGHAN-05 closed system.

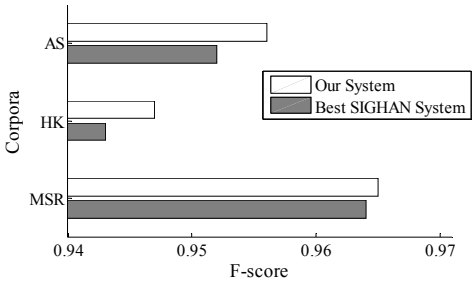

Figure. 2. Our system v.s. best SIGHAN-05

4. Analysis and Discussion

4.1. Baseline vs Best G Tagger

In comparison to the baseline, $C_{-2}C_0$, $C_{-3}C_{-1}$, are added, while C_0C_2 is removed. All the OOV recall rates are significantly improved. These bigrams can be regarded as an approximation to trigram and tetragram features. In Table 11, we list four frequent morphological rules of words whose length is longer than two. The second rule is the most frequently used among these four. Take "老人家" for example. CRF-based tagger tends to label "家" as B, because in the training corpus, "老人" is often individually tagged as a word. However, the bigram feature "C_{-2}='老' and C_0= '家' " provides information for the CRF-based tagger that the current character C_0 must

be tagged as the *I* tag. In addition, these bigrams outperforms those using trigrams or tetra grams directly. This is because the characters between the beginning character and the ending character sometimes can be replaced by other characters. Therefore, the unknown words or words in inconsistent construction may be tagged correctly.

4.2. G Tagger vs. GS Tagger.

Table 11. The ways composing a Chinese word of length greater than 2

Morphological Rule	Example
append chars before an existing word	市"政府" (city "government")
append chars following an existing word	"老人"家 ("elder" person)
interleave a word with another	"裝"模"作"樣 (to attitudinize)
three-character word	"對不起"(sorry)

Tagging non-Chinese strings separately can effectively block the influence of adjacent Chinese characters. This is because in most cases, the adjacent Chinese characters are independent of the tags of characters in the non-Chinese strings. Take the phrase "ＬＣＤ面板" for example. If we only use one tagger to process it, since "面" usually appears as the second character in a word of length two, the CWS tagger tends to determine "Ｄ面板" as a word, even though the relationship of "Ｄ" and "面" is very weak. Using the two-tagger method can avoid this.

Another advantage brought by is the effectiveness in resolving the data sparseness problem of bigrams that are composed of a non-Chinese character and a Chinese character. In most cases, pure Chinese bigrams have comparatively more samples, therefore, CWS performs better in pure Chinese sentences than in mixed sentences. However, since the frequencies of non-Chinese characters are lower than Chinese characters, the bigrams composed of a Chinese character and a non-Chinese character is usually unknown. Normalizing characters in the same character cluster to the same symbol can effectively resolve data sparseness of these bigrams. Take the non-Chinese string "１１比９" (11:9) for example. In the training corpus, the bigram "C_{-1}='１' and C_0= '比' " is unseen in the training set. Although other instances in the same form such as "２比２" has appeared in the training corpus, the one tagger method cannot correctly segment this string. On the other hand, by normalizing these numbers to one unused symbol, the GS tagger can correctly segment it as "１１ 比 ９."

4.2. The Effects of TBPP

The main contribution of TBPP is that it can correct long NE errors. In addition to long NEs, TBPP is also effective for identifying longer words and their variations. For example, TBPP generates the template "＊＊・＊％" from "５２・４％" and "７０・２％. " When applying this template to modify our CWS's result, "九十六 ・九％" can be fixed to "九十六・九％".

5. Conclusion

First, we add two *n*-gram features to CRF-based CWS that can effectively identify unknown words (2.6%) while achieving performance comparable with the best CWS systems. Second, we apply the K-means algorithm to character clustering for our GS tagger. This significantly improves not only the handling of sentences containing non-Chinese words but also the overall performance. Third, we develop a post-processing method that compensates for the weakness of ML-based CWS on long words and NEs. In AS, our performance is comparable to the best system (F=0.956) in the open task.

6. Acknowledgement

This research was supported in part by the thematic program of Academia Sinica under Grant AS 95ASIA02, the National Science Council under Grant NSC 95-2752-E-001-001-PAE.

7. References

[1] H. Tseng, P. Chang, Galen Andrew, D. Jurafsky, and C. Manning, "A Conditional Random Field Word Segmenter for Sighan Bakeoff 2005," presented at SIGHAN-05, 2005.

[2] T. Emerson, "The Second International Chinese Word Segmentation Bakeoff," presented at SIGHAN-05, 2005.

[3] J. Lafferty, A. McCallum, and F. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," presented at ICML-01, 2001.

[4] L. Rabiner and I. A. W. a. K.-F. Lee, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," in Readings in Speech Recognition.

[5] J. A. Hartigan and M. A. Wong, "A K-means Clustering Algorithm," Applied Statistics, vol. 28, pp. 100-108, 1979.